AD-A117 999 SACRAMENTO AIR LOGISTICS CENTER MCCLELLAN AFB CA SERV--ETC F/G 22/2
RESULTS OF A SURVEY SOFTWARE DEVELOPMENT PROJECT MANAGEMENT IN --ETC(U)
DEC 79 R H THAYER.
UNCLASSIFIED SM-ALC/MME-TR-79-54-VOL-3 NL

1 OF 2
AD A
117999

2

SM-ALC/MME TR 79-54, Volume III
18 December 1979

RESULTS OF A SURVEY SOFTWARE DEVELOPMENT PROJECT MANAGEMENT IN THE U.S. AEROSPACE INDUSTRY

VOLUME III

MAJOR PROBLEMS

RICHARD H. THAYER
SACRAMENTO AIR LOGISTICS CENTER
AIR FORCE LOGISTICS COMMAND
MCCLELLAN AFB, CA 95652

Any opinions expressed in this report are solely those of the Author and do not necessarily reflect the position of the United States Air Force or the American Institute of Aeronautics and Astronautics Technical Committee on Computer Systems.

DEPARTMENT OF THE AIR FORCE
HEADQUARTERS SACRAMENTO AIR LOGISTICS CENTER (AFLC)
MCCLELLAN AIR FORCE BASE, CALIFORNIA 95652

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SM-ALC/MME TR-79-54 VOLUME III	2. GOVT ACCESSION NO. AD-A117 999	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) RESULTS OF A SURVEY: SOFTWARE DEVELOPMENT PROJECT MANAGEMENT IN THE U.S. AEROSPACE INDUSTRY VOLUME III: MAJOR PROBLEMS		5. TYPE OF REPORT & PERIOD COVERED Technical, Final 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Richard H. Thayer		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Directorate of Materiel Management Sacramento Air Logistics Center McClellan Air Force Base, California 95652		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Volume III: Major Problems
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE 18 December 1979 13. NUMBER OF PAGES 135
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, unlimited distribution.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Prepared in cooperation with the American Institute of Aeronautics and Astronautics (AIAA) Technical Committee on Computer Systems.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software Engineering Project Management, Software Development, Survey, Project Management, Major Issues

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

See separate sheet following.

DD FORM 1 JAN 73 1473

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

RESULTS OF A SURVEY
SOFTWARE DEVELOPMENT PROJECT MANAGEMENT
IN THE U.S. AEROSPACE INDUSTRY

Volume III: MAJOR PROBLEMS

BY

RICHARD H. THAYER

This report contains the results of a survey conducted in 1977 and 1978 on how the U.S. Aerospace Industry manages its software development projects. The sample of the U.S. Aerospace Industry that was surveyed consisted of companies with membership in the AIAA Technical Committee on Computer Systems. The survey contained a number of different parts. This volume pertains to Part Three and reports on how project managers perceive the major issues and/or major problems of software engineering project management. Other volumes pertain to Part One and Part Two.

Twenty major problems were postulated and those surveyed were asked to comment on whether or not they felt the problems cited were critical, important, not important, or no problem at all. In addition the respondents were asked whether or not the problem was management, technical, both, or neither; and whether or not the problem could be solved through improvements in management, technology, both, or neither. Lastly, the surveyees were asked to describe how they would (or did) solve the problem.

The answers have been condensed and/or coded and recorded on a tabulation sheet in this report. In addition, the narrative portion of the survey is recorded in clear text with all references to individuals and/or their companies deleted. This report does not attempt to analyze or come to conclusions about the data, only to report it as clearly as possible.

TABLE OF CONTENTS

SECTION 1
RESULTS

BACKGROUND

In the spring and summer of 1977, a survey was conducted on the U.S. Aercspace Industry to determine what management techniques and procedures they were employing in software development projects. This survey was designed, written, tested and implemented by the author and Mr. John H. Lehman, California State University, Sacramento, California. It was originally accomplished to collect data for analysis and the preparation of a paper on Software Engineering Project Management, to be presented at the American Institute of Aeronautics and Astronautics (AIAA) Conference, Computers in Aerospace, 31 Oct-2 Nov 1977.

The sample of the U.S. Aerospace Industry surveyed consisted of those firms and companies with a membership in the AIAA Technial Committee on Computer Systems who were hosts to the conference. These committee members represented 47 major corporations, or major corporate subdivisions, and occupied top positions in software management within their firms. They were, therefore, in an ideal position to report on how their segment of the U.S. Aerospace Industry managed its software development projects.

Initial contact was made in May 1977 to determine which members of the committee would be interested, willing, and able to participate. Forty-five members, representing 35 companies, agreed to respond. The initial draft of the survey was completed in June 1977 and critiqued by approximately 25% of the total committee membership. The results of this critique, along with other corrections, were incorporated into the final survey. The survey was mailed 10 August 1977. On 6 September 1977, with 29 of the completed surveys on hand, the authors of the survey wrote the first report for the proceedings of the Conference, Computers in Aerospace. This paper can be found in the Conference Proceedings, A Collection of Technical Papers. By the time the actual presentation was given on 1 November 1977, questionnaires from 33 companies representing 55 projects had been received. These companies, predominantly aerospace firms with government contracts, reported mostly on large to very large projects. The presentation given (called Report Nr 2, AIAA Project Management Survey) differed from the paper in so far as it used the more complete data and a different approach.

The survey did not end there, for completed forms continued to arrive until, by the summer of 1979, 66 projects representing 38 firms for a 86% return rate had been received (see Appendix A for a list of participants). A decision was made by the AIAA Technical Committee on Computer Systems to make further use of the data by writing an assessment paper on the state-of-the-art in software development project management. Mr Gene F. Walters, General Electric Co., Command and Information Systems, Sunnyvale, California and Mr Jack E. Bloodworth, Boeing Aerospace Company, were given primary responsibility for this paper. In addition, the Rome Air Development Center (RADC), the Boeing Aerospace Company and the Sacramento Air Logistic Center offered their services, and in some cases the services of their company's data processing capability to reduce and analyze the data.

The remaining problem was to reduce the data into a form useable by a computer. This involved "coding" the narrative and free form answers of the survey and verifying their consistency.

PURPOSE OF SURVEY

As previously stated, the purpose of this survey was to look at a sample of the U.S. Aerospace Industry through the use of a questionnaire to determine how they managed software development projects. Specifically, the questions that the survey attempted to answer were:

1. What are the current practices in Software Engineering Project Management today?

2. Are the new developments in management, i.e., "modern" management techniques or project management techniques, being used?

3. What are the trends in Software Engineering Project Management?

4. What are the relationships between Software Engineering Project Management techniques and successful delivery of software?

5. What are the relationships between various parts of Software Engineering Project Management as a system?

6. What are the relationships between "modern" Software Engineering techniques and Software Engineering Project Management?

THE SURVEY

The approach taken in determining answers to these questions was to first design a model for software engineering project management as a system, and define the elements of that model and the relationships between these elements, and second develop a questionnaire around this model using the various elements and/or variables as questions and possible responses. The survey contained 225 numbered questions and by use of "questionnaire packing techniques," allowed for approximately 1,328 separate responses.

The survey, which contained 72 pages, was divided into three parts. Part One dealt with defining the total organization, management structure, requirements, and philosophy of the firm and was intended to be answered by top management to provide the backdrop against which the individual projects would be viewed. Part Two concerned questions about individual projects and was intended to be completed by the project manager. Part Three consisted of general questions, not project specific, calling for evaluation, opinions, and suggestions on the major problems of software engineering project management. It was also intended to be completed by a project manager.

PURPOSE OF THIS REPORT

This paper has been prepared to report the answers to Questions 5 through 24 of Part Three of the questionnaire in "raw" form so that they may be entered into a computer data base as well as to satisfy the many requests received from the computing community for access to the data collected as a result of this survey. The answers to Part One, Part Two and Questions 1, 2, 3, 4, and 25 of Part Three are provided in Volumes I and II. However, data from Part One and Part Two in sufficient detail to identify the position held by the respondent, as well as provide selected attributes of both the individual and the firm, have been repeated in this part to make it independent of the other two volumes. This data is reported as Psuedo Questions 1 through 4, even though these questions were not originally part of Part Three of the survey.

Because of the restrictions placed by the participants on the use of their submissions, the actual completed surveys cannot be distributed and have been destroyed. This report was selected as a means of documenting and capturing as much of the "raw data" as possible without any possibility of revealing its source. In essence, this report does not contain "raw data" but reduced data in abbreviated and coded form that efficiently separates it from its source while allowing interested computer scientists its use for their own requirements.

This report does not attempt to analyze or come to conclusions about the data, only to report it as clearly as possible. Only minimum interpretation was made to enable the answers to be tabulated for eventual analysis. Although 66 projects were reported, the author removed six projects that did not seem to fit the norm, leaving a set of 60 projects to be reported.

CONTENTS OF THIS REPORT

As already stated, the purpose of this report is not to analyze the data from the AIAA Project Management Survey, but to report it as simply and accurately as possible, and, to keep within the original ground rules of maintaining anonymity of the participants. Section 2 contains the questions and answers to this survey and Section 3 contains cited references. The participants in the survey are listed in Appendix A.

A duplicate copy of the questionnaire is in Appendix B. The questionnaire is included to allow the reader easy access to the questions and predefined answers to provide a ready familiarity with the type of material covered.

Appendix C contains the abbreviations used in reporting the narrative portions of this survey. Since the reduction of comments to code destroyed some of the richness of prose, the author felt it worthwhile to include the actual responses and these are recorded in Appendix D. To maintain the concept of protecting the participants identity, the narrative answers cannot be tied to any project reported in Section 2.

THE FUTURE

This survey is, as far as the author can determine, the first attempt to query an industry on such a large scale to discover how their software engineering projects are managed. A look at the list of contributors in Appendix A will attest to the significance of this base of answers. The tremendous volume of data collected and the excellence of the responses dictates that this store of information be made available as reference material for papers, reports, texts, and other technical publications which might benefit the U.S. Aerospace Industry or the data processing community at large. The AIAA Technical Committee on Computer Systems is anticipating the preparation of an assessment paper on industries management of software engineering projects. This committee welcomes suggestions from the computing and aerospace communities on how to best use this data for the benefit of all. Suggestions should be sent to either:

Mr Gene F. Walters
Mgr, Software Technologies
Information Systems Programs
General Electric Company
1277 Orleans Drive
Sunnyvale, CA 94086
(408) 734-4980

Mr Jack E. Bloodworth
Mgr, ALCM Software
The Boeing Aerospace Company
MS-45-70
P.O. Box 3999
Seattle, WA 98124
(206) 655-6718

The Rome Air Development Center (RADC) has contracted with ITT Research Institute (IITRI) to establish and operate a software information analysis center. The center has been named the Data and Analysis Center for Software (DACS). One of the functions of DACS is to acquire and analyze data gathered during the various phases of the software development process with the purpose of identifying and quantifying those factors which contribute to the production of quality software. The data from this survey has been contributed to DACS and is available for analysis by any member of the AIAA Technical Committee on Computer Systems as well as the general computer community. Personnel interested in receiving copies of this data, or requesting analysis of this data should contact:

Ms Lorraine Duvalle
Data & Analysis Center for Software
RADC/ISI
Griffiss AFB, NY 13441
(315) 336-0937

ACKNOWLEDGEMENTS

In addition to the contributors listed in Appendix A, the author wishes to acknowledge the support and dedication of the following people:

From the Sacramento Air Logistics Center

Personnel who provided programming and analyst support are: Ms Bonnie J. Nieland, Mr Lee M. Hanger, Mr Robert D. Heckler, Mr Grover "Bob" Collins, Mr John W. Robino, and Mr David E. Sturdevant.

The following individuals provided typing, proofreading, and composing support: Mrs Terry L. Meyer, Mrs. Beryle E. McPheeters, Mrs Marianne L. Mueggenburg, Mrs Betty J. Smith and Miss Meg L. Astleford.

From the Boeing Aerospace Company

The Boeing Company's integrated logistic and systems maintenance team, consisting of Mr D. H. Wilson, Mr G. R. Herrold, and Mr W. B. Dalrymple, provided support in the areas of data reduction, data base structure, and file updating and verification. Dr Kenneth A. Hales, 1977 president of the AIAA Technical Committee on Computer Systems, provided the support of his committee in testing and completion of the questionnaires.

From the General Electric Company, Space Division

The Information Systems Program in Sunnyvale provided technical consultant support, proofreading, printing and encouragement through the services of Mr Gene F. Walters and his technical group.

From the Rome Air Development Center

RADC had offered to perform analysis of the data for the benefit of the U.S. Air Force, the AIAA Technical Committee on Computer Systems, and the computing community. Personnel responsible for this are: Mr Donald Roberts and Mr Alan R. Barnum. Ms Lorraine Duval, ITT Research Institute, who is general manager of the RADC Data and Analysis Center for Software (DACS), became the custodian of the data from this survey.

ATTACHMENT 1 TO SECTION 1

RELATIONSHIPS BETWEEN REPORTS

The survey was comprised of three parts, each dealing with a separate facet of software engineering project management. Part One dealt with the firm and the environment in which the project was done. Part Two was devoted to specific software engineering projects accomplished within the firm. Part Three asked the project managers their opinions about project management. Each of these parts can stand alone. Part One, covered in Volume I of this report series, centers on the organization, management policies, staffing techniques and project controls of the companies that completed project questionnaires reported in Part Two.

Part Two, reported in Volume II of this report series, provided both detail and summary information on each project for which a valid questionnaire was returned. Each questionnaire could be considered a case study in project management. Part Three, reported in this Volume, concerns ideas and perceptions about software engineering project management but does not relate to a given project or company.

At the same time, there is a relationship between these reports. Table 1 tells the relationships between Volumes I, II and III of this report.

TABLE 1 (ATTACHMENT 1 TO SECTION 1)
RELATIONSHIPS OF PROJECTS REPORTED IN AIAA
PROJECT MANAGEMENT SURVEY VOLUMES I, II AND III

Survey Identification Nr (1)	VOL I (Part One) (2)	VOL II (Part Two) (3)	VOL III (Part Three) (4)
101	30	101	Yes
102	30	102	Yes
103	30	103	Yes
104	31	104	Yes
105	33 (8)	105	Yes
106	34 (8)	106	Yes
107	35	107	Yes
108	35	108	Yes
109	35	109	Yes
110	36	110	Yes
111	36	111	Yes
112	39 (9)	112	Yes
113	40 (9)	113	Yes
114	41	114	Yes
115	69	115	No
116	42	116	None
117	43	117	Yes
118	45	118	Yes
119	45	119	Yes
120	51	120	Yes
121	66 (5)	121	Yes
122	51	122	Yes
123	51	123	Yes
124	51	124	Yes
125	52	125	Yes
126	55	126	Yes
127	None	127	Yes
128	59	128	No
129	None	129	Yes
130	31	130	Yes

(TABLE 1 CONTINUED)

Survey Identification Nr (1)	VOL I (Part One) (2)	VOL II (Part Two) (3)	VOL III (Part Three) (4)
201	67	201	None
202	27 (7)	202	Yes
203	28 (7)	203	Yes
204	29	204	Yes
205	32	205	Yes
206	37	206	Yes
207	37	207	Yes
208	38	208	Yes
209	43	209	Yes
210	44	210	Yes
211	46 (10)	211	Yes
212	47 (10)	212	Yes
213	49	213	Yes
214	49	214	Yes
215	49	215	Yes
216	49	216	Yes
217	50	217	Yes
218	53 (11)	218	Yes
219	54 (11)	219	Yes
220	56	220	Yes
221	57	221	Yes
222	60	222	Yes
223	60	223	Yes
224	58	224	Yes
225	58	225	Yes
226	58	226	Yes
227	61	228	Yes
228	61	228	Yes
229	64	229	Yes
230	68	230	Yes
301	26 (6)	301	Yes
302	48 (10)	None	None
303	25 (5)	None	None
304	68	304 (12)	None
305	62	None	None
306	63 (7)	None	Yes

FOOTNOTES FOR TABLE 1

(1) Column 1 - This column lists the returned surveys according to a randomly assigned identification number.

(2) Column 2 - The company identification number listed in column 2 is used in Vol I. In some cases, the same company was reported on by two or more individuals which resulted from two or more project managers reporting on different projects within the same company. In most instances these "double" reports were the same. Comments along these lines are contained in foot notes (5) through (12).

(3) Column 3 - This column lists the project numbers reported in Vol II. Projects with the same company numbers are from the same company or major subdivision.

(4) Column 4 - Vol III reports on data from Part Three. This column indicates whether or not the same person reported/wrote Part Two and Part Three of the survey. This is done so that the reader will know if there is any relationship between the project reported on in Part Two and the surveyee's opinions on the major problems of software development management.

(5) Company 25 and 66 are the same.

(6) Very small company. Part Three is reported as Part Three of Project 201.

(7) Company 27, 28 and 63 are the same. Answers reported under company 28 looked to be the most accurate and complete. Part Three submitted by company 63 is reported as Part Three of Project 116.

(8) Company 33 and 34 are the same. Answers reported under company 33 looked to be the most accurate and complete.

(9) Company 39 and 40 are the same and have identical answers.

(10) Company 46, 47 and 48 are the same. Answers reported under company 46 are considered to be the official answers by the surveyee.

(11) Company 53 and 54 are the same. Answers reported under company 54 looked to be the most accurate and complete.

(12) Project reported under project 304 was too large to be included.

SECTION 2
THE DATA

INTRODUCTION

This section reports the data submitted by the participants in tabulated, abbreviated, and coded form. Every effort has been made to preclude associating any response with a particular contributor.

Each question is handled separately and reported in an array format. The horizontal indices of the array refer to project identification numbers while the vertical indices refer to the question and part number. Each narrative response has been converted to a three-character alphanumeric code (see Appendix C for explanation of codes).

Questions 1 through 4 were not part of the original questionnaire, but were derived from Parts One and Two in order to make this Volume independent of Volumes I and II. Questions 5 through 24 provide the data on the 20 postulated major problems.

Part Three of Project 301 was reported as Part Three of Project 116 and Part Three of Project 306 was reported as Part Three of Project 116.

A FINAL NOTE:

The question numbers for Part Three of the questionnaire have been renumbered from 1 to 24 to 301 to 324. This was done in order to not duplicate the question number in Parts One (Volume I) and Two (Volume Two), allowing the data files to be merged.

```
QUESTION 501                    WHAT IS YOUR POSITION WITHIN YOUR COMPANY (INCLUDING UNIVERSITY
                                GOVERNMENT, ETC.) AND/OR THE COMPUTING INDUSTRY (INCLUDING
                                CONSULTANTS, STUDENTS, ETC.)?
                                A. LINE MANAGER GENERALLY OVER SOFTWARE DEVELOPMENT (INCLUDING
                                   PRESIDENT, VP, DIRECTORS, SOFTWARE DEVELOPMENT DIVISIONS, MGR, ETC)
                                B. PROJECT MANAGER RESPONSIBLE FOR THE SUCCESSFUL DELIVERY OF PROJECT
                                   (INCLUDING PROJECT MANAGER OF SOFTWARE SUBSYSTEMS)
                                C. INDIVIDUAL DEVELOPERS GENERALLY CONNECTED WITH A PROJECT (INCLUDING
                                   DESIGNERS, ANALYSTS, ENGINEERS, PROGRAMMERS, ETC)
                                D. SENIOR STAFF POSITION RESPONSIBLE FOR ESTABLISHING BROAD SOFTWARE
                                   POLICY FOR THE ORGANIZATION
                                E. SUPERVISORY/SENIOR STAFF POSITION, SOFTWARE FUNCTION (INCLUDES
                                   MISCELLANEOUS SOFTWARE DEVELOPMENT, SOFTWARE R&D, SOFTWARE
                                   TECHNOLOGY, SOFTWARE IV&V, SOFTWARE EVALUATION, ETC)
                                L. CONSULTANT
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A                                                                                                    YES
  01 B    YES YES     YES YES YES     YES YES YES YES YES YES YES YES     YES YES YES YES YES YES YES YES         C02 YES YES YES
  01 C            YES             YES
  01 D                                                                YES
  01 L                                                                                                        YES

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    YES
  01 B                    YES YES YES YES YES YES YES YES YES YES YES YES YES YES     YES YES YES     YES C01 C01 YES YES
  01 C            YES                                                                                 YES                         YES
  01 D        YES                                                             YES
  01 E                C03                                                                                                     YES
```

```
QUESTION 302                 WHICH OF THE FOLLOWING ATTRIBUTES APPLY TO THE SURVEYEE?
                             A. R&D ORIENTATED
                             B. EDUCATOR (ALL)
                             E. PROGRAMMER AND/OR SOFTWARE ANALYSTS
                             F. ENGINEER AND/OR FUNCTIONAL ANALYSIS
                             G. QUALITY ASSURANCE/TECHNICAL DIRECTOR
                             H. MANAGER/SUPERVISOR
                             I. GOVERNMENT EMPLOYEE
                             J. PHD
                             K. CONSULTANT
                             L. ESTABLISHES/INFLUENCES BROAD POLICY ON SOFTWARE DEVELOPMENT
                             M. NATIONAL AUTHOR AND/OR SPEAKER ON SOFTWARE DEVELOPMENT
                             N. AFFILIATED WITH IEEE - CS, ACM, OR OTHER DATA PROCESSING
                                PROFESSIONAL ORGANIZATIONS
                             O. AFFILIATED WITH AIAA OR OTHER AEROSPACE PROFESSIONAL ORGANIZATIONS
                             R. AMERICAN-CANADIAN INFLUENCE
            RESPONDOR
PART/SUB    101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 G                                                                                  YES
  01 H      YES YES YES YES YES YES     YES YES YES YES YES YES YES     YES YES YES YES YES YES YES YES YES YES YES YES     YES YES
  01 I                  YES                                                                                 YES     YES         YES
  01 J          YES                                             YES YES YES YES YES YES                         YES YES YES
  01 O      YES YES YES     YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES
  01 R      YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES

            RESPONDOR
PART/SUB    201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A      YES                                                                                                             YES
  01 B      YES
  01 E                                                                                                                          YES
  01 F                                                                          YES             YES
  01 H      YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES     YES YES YES     YES YES YES YES YES YES
  01 I          YES YES                             YES YES                                                                     YES
  01 J      YES                                     YES YES
  01 K      YES
  01 M      YES
  01 N      YES
  01 O          YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES
  01 R      YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES
```

```
QUESTION 305                   THE AFFILIATE, EMPLOYER OR FIRM OF THE SURVEYEE IS PRIMARILY
                               ENGAGED IN:
                               B. A MANUFACTURER OF OTHER THAN COMPUTER HARDWARE
                               C. A SOFTWARE HOUSE
                               D. AN ENGINEERING SERVICE AND TECHNICAL SUPPORT ORGANIZATION
                               E. THE GOVERNMENT: FEDERAL (NON-MILITARY), FEDERAL (MILITARY), STATE,
                                  COUNTY, MUNICIPAL
                               F. A UNIVERSITY, R&D LABORATORY, EDUCATIONAL INSTITUTION
                               M. THE UTILITIES: COMMUNICATIONS, ELECTRIC, GAS
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 B                    YES YES             YES YES YES YES YES YES     YES YES YES YES     YES YES YES             YES
  01 C    YES YES YES                                                                     YES
  01 D                            YES YES YES                                                                 YES         COS
  01 E                YES                                             YES                                 YES     YES         YES

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 B                    YES         YES YES YES         YES YES YES YES     YES YES YES     COS COS
  01 C    YES                                                             YES                                     YES YES
  01 D                YES     COS COS                                                                                     YES
  01 E        YES YES                         YES YES                                             YES YES YES             YES
  01 M                                                                                YES
```

QUESTION 304

WHICH OF THE FOLLOWING ATTRIBUTES APPLY TO THE SURVEYEE'S AFFILIATE, EMPLOYER, OR FIRM?
THE GROSS REVENUES (OR BUDGET) FOR LAST YEAR WERE:
A. LESS THAN 50 MILLION DOLLARS
B. BETWEEN 50 MILLION AND 500 MILLION DOLLARS
C. IN EXCESS OF 500 MILLION DOLLARS
WHAT PERCENT OF REVENUE IS DERIVED FROM OR BUDGET ALLOTED TO SOFTWARE DEVELOPMENT?
D. VERY LITTLE (LESS THAN 10 PERCENT)
E. MODERATE TO AVERAGE (10 TO 50 PERCENT)
F. HIGH (50 TO 90 PERCENT)
G. ALMOST ALL (GREATER THAN 90 PERCENT)
HOW MANY PEOPLE:
J. ARE EMPLOYED BY FIRM
K. WORK IN ALL ASPECTS OF SOFTWARE
L. ARE DEVOTED TO SOFTWARE DEVELOPMENT ACTIVITIES
Z. COMMENT

RESPONDOR

PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	YES	YES	YES		YES	YES	YES	YES	YES												YES			C03	YES					
01 B										YES	YES	YES	YES	YES	YES	C04		YES	YES	YES		YES	YES			YES		YES	C05	
01 C				YES																							C03			YES
01 D				YES			YES	YES	YES	YES	YES			YES	YES	YES	YES	YES	YES											YES
01 E					YES	YES						YES	YES			YES				YES		YES	YES	C03	YES	YES		YES		
01 F	YES	YES	YES																										C03	
01 G																					YES									
01 J	MIS	MIS	MIS	133	211	211	102	102	102	162	162	502	502	242	902	201	503	102	102	123	601	123	123	MIS	172	252	MIS	303	MIS	133
01 K	MIS	MIS	MIS	451	101	101	620	620	620	151	151	251	251	MIS	401	MIS	102	101	101	162	MIS	162	162	MIS	451	102	MIS	352	MIS	451
01 L	MIS	MIS	MIS	300	250	250	310	310	310	151	131	151	151	500	231	200	401	200	200	122	451	122	122	MIS	201	301	MIS	202	MIS	300

RESPONDOR

PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	YES			YES																										
01 B		C04	YES			C04	C04	C04									YES	YES	YES	YES	YES			YES	YES	YES	YES	YES	YES	
01 C					YES				C04	YES	YES	YES	YES	YES	YES	YES						YES	YES							YES
01 D					YES			YES	YES	YES			YES	YES	YES	YES		YES	YES		YES	YES	YES	YES	YES	YES				YES
01 E	YES	YES	YES			C04	C04				YES									YES							YES	YES	YES	
01 F				YES								YES					YES													
01 J	100	201	201	351	403	182	182	193	503	603	503	503	203	203	203	203	252	502	502	502	142	353	353	402	402	402	152	152	302	902
01 K	100	MIS	MIS	251	101	500	500	152	102	202	502	502	351	351	351	351	202	101	101	800	101	202	202	132	132	132	122	122	102	202
01 L	100	200	200	151	501	151	151	801	401	401	252	252	151	151	151	151	152	750	750	250	500	102	102	700	701	701	201	201	801	601

QUESTION 305

PERFORMANCE (REQUIREMENT) SPECIFICATIONS ARE FREQUENTLY INCOMPLETE, AMBIGUOUS, INCONSISTENT, MACHINE DEPENDENT, AND INVALID. (ADDED)

```
A. THIS PROBLEM IS:
   CRITICAL      (1)    AN IRRITANT          (3)
   IMPORTANT     (2)    OF NO CONSEQUENCE    (4)
   PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
B. THIS IS A PROBLEM IN:
   MANAGEMENT (1)   BOTH      (3)
   TECHNOLOGY (2)   NEITHER   (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
   MANAGEMENT (1)   BOTH      (3)
   TECHNOLOGY (2)   NEITHER   (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

	RESPONDOR																													
PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	001	002	001	003	001	001	003	002	001	001	001	001	001	001	001		001	002	002	001	001	002	001	001	002	001	001	001	002	001
01 B	003	001	003	001	003	001	001	004	001	001	001	003	003	001	003		003	003	003	003	003	003	001	003	001	001	003	001	001	001
01 C	003	001	003	001	003	001	001	004	001	003	003	003	003	001	003		003	003	003	003	003	003	001	003	001	001	003	001	001	001
01 D1	SAI	RMC	RTC	RTA	C02	RTC	RTD	RTG	RMG	RTA	RTA	RMG	RMG	N/C	RTA		RMG	RTI	PRB	RMG	RTI	RMG	RMB	C02	RMG	RMF	C02	SAI	RTE	RMC
01 D2			RMD		RTI	RMC	RTE			RME	RMF						RTC			RMA				RTI			SAJ	RTD	SAI	
01 D3							RMG													RMD				RMG			RTB	RTF		
01 D4																												RTG		
01 X																MIS														

	RESPONDOR																													
PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	001	001	002	001	002	002	001	001	002	001	001	002	001	002	001	003	002	001	001	001	002	001	001	002	001	002	002	002	001	002
01 B	003	003	002	001	003	003	002	001	003	003	002	001	003	003	001	004	003	003	001	001	001	003	003	003	001	003	003	003	003	003
01 C	003	003	002	001	001	003	002	001	001	003	002	001	002	003	001	004	003	003	001	001	001	003	003	003	001	003	003	003	003	003
01 D1	PER	RMB	C03	RME	RME	RMG	RTC	RMC	RMA	RMG	RTC	RME	RTA	RTD	RMA	RMB	RMB	RMA	RMB	N/C	N/C	RMG	RTH	RMA	N/C	RMA	N/C	RTA	RMA	RMC
01 D2		RMC	RTB						RTH			SAJ		RMB	RTD				RTA			RTI				RTI				
01 D3												RMC																		

```
QUESTION 306                    THERE ARE NO DECISION RULES FOR THE SOFTWARE ENGINEERING
                                PROJECT MANAGER TO USE IN SELECTING THE SOFTWARE DESIGN TECHNIQUES
                                OR TOOLS AVAILABLE WITHIN THE STATE-OF-THE-ART.
                                A. THIS PROBLEM IS:
                                   CRITICAL      (1)     AN IRRITANT         (3)
                                   IMPORTANT   (2)     OF NO CONSEQUENCE  (4)
                                   PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                                B. THIS IS A PROBLEM IN:
                                   MANAGEMENT (1)    BOTH       (3)
                                   TECHNOLOGY (2)    NEITHER   (4)
                                   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                                C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                                   MANAGEMENT (1)    BOTH       (3)
                                   TECHNOLOGY (2)    NEITHER   (4)
                                   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                                D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    003 004 003 003 002 002 003 002 002 001 001 002 002 001 001     001 002 002 002 001 003 002 001 001 002 002 002 005 001
  01 B    003 004 001 003 002 003 003 003 001 001 001 003 003 003 001     003 002 002 003 003 001 001 003 003 003 002 003 005 003
  01 C    003 004 004 003 002 003 003 003 001 001 001 003 003 003 001     003 002 002 003 003 001 001 003 003 003 002 003 005 003
  01 D1   N/A N/C NPR N/C XMD XTA XTA XMG SAK XMD XMD XMD C03 N/C SAJ     SMB XTC XTC XMC N/C XMD C04 N/C XTA XMD SAF XME C03 SAJ
  01 D2           NOS         XIB SAK                 XIC N/C             XME         XMB         DIS         SMB     XTC SAJ
  01 D3                       XMF                                                                 SAJ         XTA
  01 D4                                                                                                       XME
  01 X                                                            MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    001 003 003 003 004 002 005 004 002 004 004 001 003 003 004 001 003 002 003 002 002 002 002 004 002 002 003 002 003 002
  01 B    003 003 002 003 004 003 005 005 001 005 004 003 004 002 005 003 002 003 003 002 001 003 001 001 002 001 002 002 004 003
  01 C    003 001 002 003 004 003 005 005 002 005 004 003 004 002 005 003 002 003 003 002 003 003 001 001 003 001 002 002 004 003
  01 D1   PER XMD C03 XID N/C XME C03 C03 NPR C03 N/C SAJ C03 OTH C03 C03 XTF XTC N/C N/C XMA XMD XMD XMZ N/C XMA N/C XMZ SAJ XME
  01 D2       SAJ XMG             CFU DIS     N/C         XTE     XMZ SMB                         XTF     SAF
  01 D3                               XID                             PMQ                         XMF
```

```
QUESTION 307                 THERE IS NO MEASURE OR INDEX OF "GOODNESS" OF CODE THAT
                             CAN BE USED AS AN ELEMENT OF SOFTWARE DESIGN, AND THERE IS NO PRACTICA
                             WAY TO GUARANTEE ONE PROGRAM IS BETTER THAN ANOTHER.
                             A. THIS PROBLEM IS:
                                CRITICAL     (1)   AN IRRITANT        (3)
                                IMPORTANT  (2)     OF NO CONSEQUENCE  (4)
                                PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                             B. THIS IS A PROBLEM IN:
                                MANAGEMENT (1)   BOTH      (3)
                                TECHNOLOGY (2)   NEITHER   (4)
                                NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                             C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                                MANAGEMENT (1)   BOTH      (3)
                                TECHNOLOGY (2)   NEITHER   (4)
                                NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                             D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    002 004 004 003 002 002 001 003 004 002 002 004 002 003 002     002 003 003 002 002 003 005 002 002 002 003 004 002 002
  01 B    003 004 004 003 002 003 002 002 002 002 002 003 002 003 002     002 002 002 003 002 003 005 003 001 002 005 004 002 003
  01 C    003 004 004 003 002 003 002 004 002 003 003 002 002 003 002     002 002 002 003 002 002 005 003 001 002 005 004 002 003
  01 D1   IMA N/C C03 N/C IIA IID C03 N/C IMY SAI SAI IMY IMY C03 SAK     IID NOT NOT IID N/C N/C C03 N/C ITD IMY C03 IMY ITD ITD
  01 D2           NPR         XIL RES     SAK IID ITD         N/C IID                             IMY             IMY         PMW
  01 D3           IMB                                                                             SAI
  01 X                                                                MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    002 002 003 002 002 002 001 005 002 004 004 002 003 003 003 004 004 002 003 002 002 001 003 003 003 002 002 002 001 002
  01 B    003 003 003 003 003 003 002 005 003 005 003 002 002 002 002 005 004 003 003 002 002 002 002 002 004 001 003 002 003 003
  01 C    002 003 003 003 003 002 002 005 003 005 003 003 002 002 002 005 004 002 003 002 002 002 002 002 004 001 003 002 003 003
  01 D1   N/C N/C C04 PML IMY NUS IIE C03 PML C03 N/C SAI SAI SAI IMB C03 IMY ITG N/C N/C N/C XTF IMA D/K N/C PML N/C NOT ITD IMA
  01 D2           SAK                 IID ITD N/C                 ITA IMY                         XTF                         PML
  01 D3           IMA                 XIL                         PMW
  01 D4           PMW
```

QUESTION 508

```
THERE ARE NO DECISION RULES FOR SELECTING THE PROCEDURES,
STRATEGIES, AND TOOLS TO BE USED IN TESTING SOFTWARE.
A. THIS PROBLEM IS:
     CRITICAL     (1)    AN IRRITANT          (3)
     IMPORTANT    (2)    OF NO CONSEQUENCE    (4)
     PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
B. THIS IS A PROBLEM IN:
     MANAGEMENT (1)    BOTH       (3)
     TECHNOLOGY (2)    NEITHER    (4)
     NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
     MANAGEMENT (1)    BOTH       (3)
     TECHNOLOGY (2)    NEITHER    (4)
     NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

RESPONDOR

PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	002	002	003	003	002	002	002	005	002	002	MIS	002	001	001	002	002	002			001	002	005	002	002	001	002	004	002	002	002
01 B	001	002	001	005	002	003	002	002	003	001	MIS	003	003	001	005	003	003			003	002	001	002	003	005	003	005	001	003	005
01 C	001	002	004	003	002	003	002	002	003	001	001	003	003	005	005	003	003			003	002	001	002	003	003	003	005	001	005	003
01 D1	C03	N/C	C03	N/C	XMD	XTA	RES	N/C	SMB	XMF	XMF	XID	XIA	N/C	C03	C05	XID			RTB	N/C	N/C	C01	N/C	XTH	XTD	C03	XMD	XTA	XTA
01 D2	XMD		NPR			SAK									CFU								DIS					XTC	XTD	PMW
01 D3			NUS												SAJ								SAJ						XMZ	
01 X																		MIS	MIS											

RESPONDOR

PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	001	002	004	002	004	001	005	004	002	002	004	002	001	002	003	001	003	001	002	002	003	001	002	002	002	001	001	001	002	001
01 B	003	005	002	003	004	003	005	005	003	002	003	001	003	002	002	001	003	005	003	002	001	002	002	002	003	001	005	003	003	003
01 C	003	003	002	003	004	005	005	005	003	002	005	001	003	002	002	001	003	003	005	002	001	002	002	002	003	001	005	005	003	003
01 D1	N/C	XTD	C03	XTB	XTD	XTH	C03	C03	SMB	XTI	N/C	SAF	SAF	XTI	XMZ	XTI	SAF	XTI	XTA	N/C	XTI	XMA	NOS	XMZ	N/C	XMA	N/C	PRB	SAK	C02
01 D2			XMZ	XMF			DIS	XMZ	XTF						PMW															XMA
01 D3									XTJ																					
01 D4									CML																					

```
QUESTION 309                THERE IS NO MEASUREMENT, OR INDEX, OF RELIABILITY THAT
                            CAN BECOME AN ELEMENT OF DESIGN AND THERE IS NO WAY TO PREDICT
                            SOFTWARE FAILURE; I.E., THERE IS NO PRACTICAL WAY TO GUARANTEE
                            (PROVE) THE DELIVERED SOFTWARE MEETS A GIVEN DESIGN CRITERIA
                            A. THIS PROBLEM IS:
                               CRITICAL     (1)    AN IRRITANT        (3)
                               IMPORTANT    (2)    OF NO CONSEQUENCE  (4)
                               PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                            B. THIS IS A PROBLEM IN:
                               MANAGEMENT (1)    BOTH      (3)
                               TECHNOLOGY (2)    NEITHER   (4)
                               NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                            C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                               MANAGEMENT (1)    BOTH      (3)
                               TECHNOLOGY (2)    NEITHER   (4)
                               NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                            D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

RESPONDOR

PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	002	005	002	003	001	002	005	002	002	002	002	002	002	005	002	003	002			003	002	003		002	003	002	002	002	002	002
01 B	003	003	003	003	002	003	003	002	003	003	003	003	002	005	002	003	002			003	002	002		002	004	002	002	001	003	003
01 C	003	003	003	003	002	003	003	004	003	003	003	003	002	005	002	004	003			003	002	002		002	004	002	002	001	003	003
01 D1	XTF	N/C	IMY	N/C	XIL	XIL	IIJ	N/C	IIK	XIL	XIL	XIL	ITG	C03	XTF	N/C	C02			IIJ	N/C	IIJ		N/C	IMA	NOS	C03	RID	XIL	IID
01 D2	III												IIJ	N/C			NPR			III							N/C	XTF	ITG	
01 D3	PMI																											IIJ		
01 X																		MIS	MIS				MIS							

RESPONDOR

PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	005	001	004	001	002	001	001	001	002	001	004	001	002	001	002	001	002	001	003	003	001	001	002	004	002	002	002	002	002	001
01 B	005	003	005	003	003	003	002	002	002	002	003	001	002	002	003	001	002	003	002	002	003	003	002	004	003	001	001	002	003	003
01 C	005	003	005	003	002	002	002	001	002	002	003	003	002	002	003	003	002	003	002	002	003	003	002	004	003	001	001	002	003	003
01 D1	C03	NOS	C03	PML	ITJ	IMA	XTF	IMY	SAI	IIK	N/C	N/C	IIJ	IIJ	NPR	ITJ	ITC	IIJ	N/C	N/C	XTF	IIJ	N/A	ITJ	N/C	ITJ	N/C	NOT	C01	C02
01 D2	DIS	XIL	NOS		ITD		ITB		IIJ						PMW						ITJ								IID	IMY

```
QUESTION 310                THERE IS NO WAY TO GUARANTEE THAT THE DELIVERED SOFTWARE
                            MEETS THE USER'S REQUIREMENTS.
                            A. THIS PROBLEM IS:
                               CRITICAL    (1)    AN IRRITANT        (3)
                               IMPORTANT   (2)    OF NO CONSEQUENCE  (4)
                               PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                            B. THIS IS A PROBLEM IN:
                               MANAGEMENT (1)   BOTH      (3)
                               TECHNOLOGY (2)   NEITHER   (4)
                               NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                            C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                               MANAGEMENT (1)   BOTH      (3)
                               TECHNOLOGY (2)   NEITHER   (4)
                               NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                            D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?

           RESPONDOR
PART/SUB   101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A     004 004 002 001 001 001 001 001 001 001 001 001 001 001 002 002 001 002 002 002 002 001 001 002 001 003 001 001 005 001
  01 B     004 001 004 003 005 001 001 005 003 005 005 003 001 001 004 002 003 002 002 003 002 005 001 005 001 001 005 001 005 001
  01 C     004 001 004 003 005 001 001 005 003 005 003 003 003 005 001 003 003 002 002 003 002 005 001 003 001 001 003 001 005 001
  01 D1    ITJ RMG RTA NOS ITJ PMI ITD N/C ITJ IMC IMC IMA RTB N/C RES NOS C02 NPR NPR RTA IMC IMA RTA C02 N/C NOS RTB IMA C05 IMA
  01 D2                        IMA ITJ                 IMC ITJ             CTB         IMA         ITK N/C             ITG DIS
  01 D3                        ITJ                     ITJ

           RESPONDOR
PART/SUB   201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A     001 001 004 001 003 001 001 001 002 005 004 001 001 001 002 002 002 001 002 002 003 002 001 003 001 001 001 001 001 001
  01 B     005 005 005 003 001 001 002 002 001 004 003 001 001 002 001 003 001 005 002 001 001 003 003 002 001 001 001 005 001 003
  01 C     005 003 005 005 001 001 002 001 001 004 003 001 002 002 003 004 001 003 002 001 001 005 005 002 001 001 001 005 001 003
  01 D1    N/C RTB C03 RTA IMA CTB ITJ C01 IMA IMY N/C IMA C05 ITJ ITA N/C IMA IMA N/C N/C ITJ IMA RTA ITJ N/C RMG N/C IMA IMC C02
  01 D2            ITJ ITJ             IMY                 ITJ     ITJ                                 IMA         IMA         PMI IMY
```

```
QUESTION 311          THERE IS NO MEASUREMENT OR INDEX OF MAINTAINABILITY
                      THAT CAN BECOME AN ELEMENT OF DESIGN ; I.E., THERE IS NO PRACTICAL
                      WAY TO GUARANTEE THAT A GIVEN PROGRAM IS MORE MAINTAINABLE THAN
                      ANOTHER.
                      A. THIS PROBLEM IS:
                         CRITICAL    (1)   AN IRRITANT        (3)
                         IMPORTANT   (2)   OF NO CONSEQUENCE  (4)
                         PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                      B. THIS IS A PROBLEM IN:
                         MANAGEMENT (1)   BOTH     (3)
                         TECHNOLOGY (2)   NEITHER  (4)
                         NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                      C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                         MANAGEMENT (1)   BOTH     (3)
                         TECHNOLOGY (2)   NEITHER  (4)
                         NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                      D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

RESPONDOR

PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115
01 A	004	004	004	002	002	002	003	002	003	002	002	004	002	004	003
01 B	004	004	005	003	002	001	003	002	003	002	002	003	001	005	005
01 C	004	004	005	003	002	001	003	002	003	002	002	003	003	005	005
01 D1	XTM	N/C	C03	NUS	IMB	XTL	ITD	N/C	IMC	RES	RES	DIS	XTL	C03	C03
01 D2			IMB		ITD	XTL							XTM	N/C	XTL
01 D3			IMD		ITF	XTM									
01 X															

PART/SUB	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	002	002			003	002	003		003	002	002		002	002	002
01 B	003	003			003	002	002		002	003	003		002	003	003
01 C	003	003			003	002	002		002	003	003		002	003	003
01 D1	N/C	CTB			XTL	N/C	IMB		N/C	XTL	XTL		ITD	SAI	ITD
01 D2					PMW					XTM			IMB		
01 D3					XTF										
01 X			MIS	MIS				MIS				MIS			

RESPONDOR

PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215
01 A	001	002	004	002	003	002	002	002	001	003	002	002	002	002	002
01 B	003	003	005	003	004	003	001	002	003	004	003	001	002	002	003
01 C	003	003	005	003	004	003	001	002	003	004	003	001	002	002	003
01 D1	N/C	ITD	C03	PMI	NPR	PML	IMC	C03	XTF	IMB	N/C	XTL	SAB	NUS	C01
01 D2			XTF				XTF	XIF	XIF	PML					N/C
01 D3								XTM	XTM						

PART/SUB	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	004	002	002	003	002	003	002	004	003	002	002	002	001	002	002
01 B	005	003	003	001	001	001	002	004	001	002	001	003	003	003	002
01 C	005	003	003	001	001	001	002	004	001	002	001	003	003	003	002
01 D1	C03	XTF	XTL	IMY	N/C	PML	XTF	NUS	SAI	N/C	XTL	N/C	D/K	XTM	XTF
01 D2	N/C		PMW			XTL					XTM				XTM
01 D3											PML				

```
QUESTION 512              NO TECHNICAL DISCIPLINE EXISTS FOR THE DESIGN OF MAINTAINABLE
                          PROGRAMS.
                          A. THIS PROBLEM IS:
                             CRITICAL    (1)   AN IRRITANT        (3)
                             IMPORTANT   (2)   OF NO CONSEQUENCE  (4)
                             PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                          B. THIS IS A PROBLEM IN:
                             MANAGEMENT (1)   BOTH      (3)
                             TECHNOLOGY (2)   NEITHER   (4)
                             NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                          C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                             MANAGEMENT (1)   BOTH      (3)
                             TECHNOLOGY (2)   NEITHER   (4)
                             NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                          D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 12  122 123 124 125 126 127 128 129 130
  01 A    004 004 004 002 002 002 003 002 003 002 002 004 003 003 003 002 002         003 002 004     003 003 002     002 002 002
  01 B    004 004 005 001 002 001 002 002 002 002 002 003 001 005 005 003 003         003 002 005     003 002 003     002 002 003
  01 C    004 004 005 003 002 001 002 002 002 002 002 003 001 005 005 003 003         003 002 005     003 002 003     002 002 003
  01 D1   XTF N/C C03 NOS XTB XTK RES N/C N/C RES RES XTL XMD C03 C03 N/C CTB         C01 N/C C03     C02 N/C XTC     XME N/C XTA
  01 D2   XMF     XMY                                         N/C XIL                 XTL     XMZ     XIC             XID     PMW
  01 D3           XMD                                                                 PMW
  01 D4           PMW                                                                 XTF
  01 X                                                                    MIS MIS                 MIS             MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    005 002 005 001 003 002 002 004 005 002 004 002 002 002 002 001 002 003 003 003 003 002 004 005 002 002 002 001 002 002
  01 B    005 001 005 003 003 003 001 005 005 002 003 001 002 002 003 001 003 003 001 001 003 002 004 005 002 003 001 003 003 002
  01 C    005 001 005 003 003 003 001 005 005 002 003 001 002 002 003 001 003 003 001 001 003 002 004 005 002 003 001 003 003 002
  01 D1   C03 XID C03 XIF XIF XIC XTM C03 C03 XIA N/C N/C SAB C03 C03 C03 XTF PMW N/C N/C N/C XTF N/A C03 N/C XTB N/C N/C XTC XTN
  01 D2   DIS     XIA             XMF XMF N/C DIS                 N/C XIA SAB                                                 XTM
  01 D3                                                               XTB
  01 D4                                                               PMW
```

QUESTION 515

```
IT IS REPORTED THAT PLANNING FOR SOFTWARE ENGINEERING PROJECTS IS
GENERALLY POOR.
A. THIS PROBLEM IS:
   CRITICAL     (1)    AN IRRITANT          (3)
   IMPORTANT  (2)      OF NO CONSEQUENCE    (4)
   PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) 85)
B. THIS IS A PROBLEM IN:
   MANAGEMENT (1)    BOTH       (3)
   TECHNOLOGY (2)    NEITHER    (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
   MANAGEMENT (1)    BOTH       (3)
   TECHNOLOGY (2)    NEITHER    (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

RESPONDOR

PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	001	002	002	002	002	002	001	001	001	001	001	002	001	001	001	002	001	001	001	004	002	001	001	001	001	001	001	001	005	001
01 B	001	001	001	003	003	001	001	003	001	001	001	003	001	001	001	001	001	001	001	004	003	003	001	001	001	001	001	001	005	001
01 C	001	001	001	003	003	001	001	003	001	001	001	003	001	001	004	003	001	001	001	004	003	003	003	001	001	001	001	001	005	001
01 D1	PMA	N/C	PMF	N/S	C02	PML	PMP	PMI	PMH	PMB	PMB	RMB	PMA	N/C	NPR	N/C	CMG	NOT	PMA	PMA	SMK	PMK	PMD	N/C	PMR	PMA	SMK	PMA	C05	PHB
01 D2	PMM		OTH		PML		PMG		PMP			R11	PMP				PMA			PMM								PMM	DIS	SMK
01 D3	PMP						PMP					PMA					PMP			RMD										PMB

RESPONDOR

PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	001	005	003	001	003	002	002	004	001	001	001	002	001	001	001		002	001	001	001	001	002	002	002	001	001	001	001	002	002
01 B	001	005	003	003	003	003	001	005	001	003	001	001	003	003	003		001	003	004	001	001	001	001	001	001	001	001	003	001	001
01 C	003	005	001	003	003	003	001	005	001	003	001	001	003	003	001		001	003	004	001	001	001	001	001	001	001	001	003	001	001
01 D1	N/C	C03	C04	PMH	RID	PML	PMR	C03	PMA	PER	PML	N/C	PMA	SMK	PMC		PMH	PMF	DIS	N/C	SMK	PMA	PMA	SMK	N/C	PMB	N/C	PMR	PMA	OTH
01 D2		N/A	PMA		RIB		CMD	N/C	PMM						SAK							PMP				PMA		OTH	PMJ	
01 D3			PMM						PMN																	PMM				
01 X																MIS														

```
QUESTION 314                  THERE IS AN INABILITY TO ACCURATELY ESTIMATE DELIVERY TIME IN A
                              COMPUTER PROGRAM.
                              A. THIS PROBLEM IS:
                                 CRITICAL     (1)    AN IRRITANT          (3)
                                 IMPORTANT  (2)     OF NO CONSEQUENCE  (4)
                                 PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                              B. THIS IS A PROBLEM IN:
                                 MANAGEMENT (1)    BOTH       (3)
                                 TECHNOLOGY (2)    NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                              C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                                 MANAGEMENT (1)    BOTH       (3)
                                 TECHNOLOGY (2)    NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                              D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    002 001 002 003 001 002 001 001 002 001 001 001 001 001 002 002 001 002 002 004 002 001 001 001 002 002 002 001     001
  01 B    001 003 001 003 003 001 003 003 003 003 003 003 003 001 001 003 003 004 004 004 001 003 001 001 003 001 001 001     003
  01 C    001 003 001 003 003 001 003 003 003 003 003 003 003 001 004 003 003 004 004 004 001 003 001 001 003 001 001 001     003
  01 D1   PSF C02 RTD NUS RIA RTA NUS N/C RTD PSI PSI SAJ SAJ N/C PSL N/C CMG NOS NPR PMA PSJ CMD PRB N/C C02 PMR N/C PMA     PSJ
  01 D2   PMR PMR                 RIA     RIA PSH PSH RMB PSJ                 PMA         RTA                     PSH PSJ     SAD
  01 D3   PSJ PSF                 PSG         CMD CMD     PSL                 PMP                                             PSH
  01 D4   PSK                                             PSF
  01 X                                                                                                                            MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    001 002 004 002 005 002 002 004 001 001 001 002 002 002 002 001 002 001 002 002 002 002 002 002 002 001 002 001 002 002
  01 B    003 001 003 003 005 003 001 005 001 003 003 001 001 003 001 001 001 002 001 001 001 001 001 003 003 003 002 003 001 001
  01 C    003 001 003 003 005 003 001 005 001 003 003 001 005 003 001 001 001 002 001 001 001 001 001 003 003 003 002 003 001 001
  01 D1   N/C PMR C03 XTF C03 SAK PSD C03 PSA PSI N/C N/C C03 SAJ C03 C03 PMR XTF CMD N/C N/C PMR RTA DIS N/C PSA N/C CMD PMA RTD
  01 D2       PSB PSD     DIS         PMR CMD             D/K     PSY RME                             PSA RTA     PMR     PSL
  01 D3                                                                                                               PSB
  01 D4                                                                                                               PSJ
```

```
QUESTION 315                  THE ABILITY TO PLAN FOR RESOURCES, PARTICULARLY THE NUMBER OF
                              PROGRAMMERS REQUIRED IS POOR.
                              A. THIS PROBLEM IS:
                                 CRITICAL     (1)   AN IRRITANT          (3)
                                 IMPORTANT  (2)     OF NO CONSEQUENCE   (4)
                                 PROBLEM INCORRECTLY OR CONFUSING (ADDED) (5)
                              B. THIS IS A PROBLEM IN:
                                 MANAGEMENT (1)    BOTH       (3)
                                 TECHNOLOGY (2)    NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) 85)
                              C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                                 MANAGEMENT (1)    BOTH       (3)
                                 TECHNOLOGY (2)    NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) 85)
                              D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    002 004 002 001 002 002 002 002 002 002 002 002 002 001 002 002 001 002 002 004 002 002     001 001 002 002 001     001
  01 B    001 004 001 003 001 001 003 003 003 003 003 001 001 001 001 003 003 001 001 004 001 001     003 001 001 001 001     003
  01 C    001 004 001 003 001 001 003 003 003 003 003 001 001 001 004 003 003 001 001 004 001 001     001 001 001 001 001     003
  01 D1   PSK N/C PMJ NUS SAF RTA PSC N/C SAJ PSI PSI SAJ SAJ N/C SAJ N/C CMG SAG NPR PSL PMR PSF     C02 PSD SAH N/C PMA     PSJ
  01 D2           PSF         SAE OTH     SAF PSH PSH                     PMA         PMR             N/C             SAD
  01 D3                       PSF             CMD SMJ                     PMP         PSB                             PSB
  01 X                                                                                        MIS                     MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    001 002 001 001 005 002 002 004 001 002 002 002 002 002 002 005 002 001 002 003 002 003 004 003 002 002 002 002 002 002
  01 B    003 001 003 003 005 003 001 005 001 003 001 001 001 001 003 005 001 003 004 001 001 001 004 001 003 001 001 003 001 001
  01 C    003 001 003 003 005 003 001 005 004 003 001 001 005 001 001 005 001 003 004 001 003 001 004 001 003 001 001 003 001 001
  01 D1   N/C PST C02 XIF C03 SAK PSD C03 PMR NOT N/C N/C C03 SAJ C03 C03 PSI RME DIS N/C N/C PSY N/A SAK N/C PSA N/C PSL PSD PSL
  01 D2       SAJ PSD SAI DIS         PMR PSI VAR         D/K     PMA PSY                             SAH     PMR     CMD     CMD
  01 D3                                                           PSY                                         PSB
  01 D4                                                                                                       PSJ
```

QUESTION 316

THERE IS NO REAL QUALITY METHOD OF DESIGNING A PROJECT CONTROL
PLAN THAT WILL ENABLE PROJECT MANAGERS TO CONTROL THEIR PROJECT.

A. THIS PROBLEM IS:
   CRITICAL (1)   AN IRRITANT (3)
   IMPORTANT (2)   OF NO CONSEQUENCE (4)
   PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) 85)

B. THIS IS A PROBLEM IN:
   MANAGEMENT (1)   BOTH (3)
   TECHNOLOGY (2)   NEITHER (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)

C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
   MANAGEMENT (1)   BOTH (3)
   TECHNOLOGY (2)   NEITHER (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)

D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?

RESPONDOR

PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	002	004	002	002	001	001	004	002	002	001	001	002	002	004	004	004	002	002	002	004	002	003		001	001	001	002	001		001
01 B	001	004	001	001	001	001	005	001	003	001	001	001	001	005	005	004	003	001	001	004	003	001		001	001	001	001	001		003
01 C	001	004	001	001	001	001	005	001	003	001	001	001	001	005	005	005	003	001	001	004	003	001		001	001	001	001	001		003
01 D1	CMA	N/C	C01	NOS	CMD	OMA	C03	N/C	CMJ	CMJ	CMJ	CMI	N/C	C03	C03	CMF	SAJ	PMQ	PMQ	CMD	CMK	CMA		N/C	OMA	CMC	N/C	CMF		NOS
01 D2			CMI			PMA	N/C		SAI	NPR	NPR			N/C	NPR	N/C	CMD			CMK						CMD				
01 D3						PMH											SMA													
01 X																							MIS						MIS	

RESPONDOR

PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	001	005	001	001	003	005	002	001	001	002	004	001	002	002	001	005	002	002	002	002	002	002	003	004	002	001	003	002	001	002
01 B	003	005	001	003	003	005	001	001	001	003	001	001	001	001	003	005	001	003	004	001	001	001	001	004	001	001	003	001	001	001
01 C	003	005	001	003	003	005	001	001	003	003	001	001	001	001	003	005	001	003	001	001	001	001	001	004	001	003	003	001	001	001
01 D1	N/C	C03	CMA	CMB	CMD	C03	CMD	SMA	DIS	CMC	N/C	N/C	N/C	N/C	C05	C03	CMD	N/C	DIS	N/C	N/C	SAJ	NOT	NPR	N/C	RIA	N/C	SMA	PMA	NPR
01 D2		NPR		RIA		DIS									CMF	NPR														
01 D3															SAI															
01 D4															PMW															

```
QUESTION 517                  THERE ARE NO DECISION RULES FOR THE SELECTION OF MANAGEMENT
                              TECHNIQUES FOR SOFTWARE ENGINEERING PROJECT MANAGEMENT.
                              A. THIS PROBLEM IS:
                                 CRITICAL     (1)   AN IRRITANT            (3)
                                 IMPORTANT  (2)     OF NO CONSEQUENCE   (4)
                                 PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                              B. THIS IS A PROBLEM IN:
                                 MANAGEMENT (1)   BOTH      (3)
                                 TECHNOLOGY (2)   NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                              C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                                 MANAGEMENT (1)   BOTH      (3)
                                 TECHNOLOGY (2)   NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                              D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    002 004 002 002 002 003 004 003 003 002 002 004 003 004 004 004 001 002 002 004 002 003     001 003 002     002 002 001
  01 B    001 004 001 003 001 001 005 001 003 001 001 001 001 005 005 005 001 001 001 004 001 001     001 001 001     001 001 003
  01 C    001 004 001 003 001 001 005 001 003 001 001 001 001 005 005 005 001 001 001 004 001 001     001 001 001     001 001 003
  01 D1   DMA N/C DM7 NUS DME DME C03 N/C DMD DMB DMH DMD N/C C03 C03 C03 DMF PME N/S DMO N/C DMO     N/C DMB DMB     NUN SMA NUS
  01 D2           SMH             N/C                             N/C NPR N/C SMK
  01 X                                                                                        MIS             MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    001 005 003 002 003 005 003 004 004 004 002 002 002 002 004 004 004 002 002 002 003 003 003 004 002     002 004 002 002
  01 B    003 005 001 003 003 005 001 005 005 005 001 001 001 001 005 005 004 003 004 001 001 004 001 004 001     001 005 001 001
  01 C    003 005 001 003 003 005 001 005 005 005 001 003 005 001 005 005 004 003 001 001 001 004 001 004 001     001 005 001 001
  01 D1   N/C C03 C03 DMB DMC C03 N/C C03 C03 C03 N/C N/C C03 N/C C03 C03 N/C N/C DIS N/C N/C DMC DMF NPR N/C     N/C C03 DME DMB
  01 D2       NPR DME PMG     DIS     PMA DIS DMC         NUT     DIS DME                         NUT                 DMC
  01 D3                               DMZ
  01 X                                                                                                    MIS
```

QUESTION 51B

```
TECHNIQUES FOR THE SELECTION OF PROJECT MANAGERS ARE POOR,
GENERALLY RESULTING IN POORLY MANAGED PROJECTS.
A. THIS PROBLEM IS:
   CRITICAL     (1)    AN IRRITANT          (3)
   IMPORTANT    (2)    OF NO CONSEQUENCE    (4)
   PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
B. THIS IS A PROBLEM IN:
   MANAGEMENT (1)    BOTH      (3)
   TECHNOLOGY (2)    NEITHER   (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
   MANAGEMENT (1)    BOTH      (3)
   TECHNOLOGY (2)    NEITHER   (4)
   NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADED) (5)
D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

	RESPONDOR														
PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115
01 A	001	002	002	001	002	002	004	002	002	002	002	004	003	003	005
01 B	001	001	001	001	001	001	005	001	003	001	001	001	001	005	005
01 C	001	001	001	001	001	001	005	001	003	001	001	001	001	005	005
01 D1	SMH	N/C	C01	NUS	SMI	SMA	C03	N/C	SML	SMB	SMB	SMB	SMH	C03	C03
01 D2			SMH				N/C			SMI	SMI				N/C
01 D3										SML	SML				
01 X															

PART/SUB	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	002	001	002	002		002	001		002	001	002	001	001	002	001
01 B	002	001	001	001		001	001		001	003	001	001	001	001	001
01 C	004	001	001	001		001	001		001	003	001	001	001	001	001
01 D1	N/C	SMH	NUS	SMK		SMC	NPR		N/C	SMD	UNK	N/C	SMA	SMH	SMK
01 D2	SMA	SML	SMH	SMG		PRB							SAJ		
01 D3															
01 X					MIS			MIS							

	RESPONDOR														
PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215
01 A	001	005	001	001	005	005	004	004	001	005	002	001	002	MIS	004
01 B	003	005	001	003	001	005	005	005	001	001	001	001	001	001	005
01 C	003	005	001	003	001	005	005	005	001	001	001	001	001	001	005
01 D1	N/C	C03	C05	SMH	C03	C03	C03	C03	SMA	C03	N/C	N/C	OTH	N/C	C03
01 D2		DIS	SMH		DIS	DIS	DIS	N/A	SMH	DIS					PMA
01 D3									SMF						SAK

PART/SUB	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	003	002	001	002	002	001	002	004	003	001	001	002	002	002	002
01 B	001	001	003	004	001	001	001	001	001	001	001	001	001	001	001
01 C	001	001	003	001	001	001	001	001	001	001	001	001	001	001	001
01 D1	C03	SMH	N/C	DIS	N/C	N/C	SHH	SMH	NPR	N/C	N/C	N/C	SMJ	SMH	SMJ
01 D2	SML			SMH										SME	SMH
01 D3															

```
QUESTION 319                THERE IS NO MEANS OF MEASURING WITH ANY DEGREE OF ACCURACY THE
                            QUALITY OF CODE PRODUCED BY A PROGRAMMER.
                            A. THIS PROBLEM IS:
                               CRITICAL    (1)    AN IRRITANT          (3)
                               IMPORTANT   (2)    OF NO CONSEQUENCE    (4)
                               PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                            B. THIS IS A PROBLEM IN:
                               MANAGEMENT (1)    BOTH      (3)
                               TECHNOLOGY (2)    NEITHER   (4)
                               NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                            C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                               MANAGEMENT (1)    BOTH      (3)
                               TECHNOLOGY (2)    NEITHER   (4)
                               NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                            D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDUR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    002 004     001 002 002 002 002 003 002 002 002 002 003 004 003 002 002 002 003 002 003     002 003 002 004 002 002 002
  01 B    002 004     003 003 001 003 002 003 003 003 003 003 005 005 002 002 002 002 003 002 001     003 001 001 005 001 003 003
  01 C    002 004     003 003 001 003 002 003 003 003 003 003 005 005 002 002 002 002 003 002 001     001 001 001 005 001 003 003
  01 D1   C03 N/C     NUS XII XIL SAE N/C NPR CTI CTI SAA SAA C03 C03 N/C SMH XIF XIF XIF N/C SAA     C02 SMA N/C C03 SAA RES ITD
  01 D2                       SAA SAA     SAD         CTG     N/C CMD     SML         IIJ             N/C         N/A CMG SAE PMW
  01 X            MIS                                                                             MIS

          RESPONDUR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    005 002 005 002 002 002 002 004 002 002 002 001 002 002 003 001 003 001 002 002 002 003 002 003 002 001 002 004 001 003
  01 B    005 003 005 003 001 003 002 005 003 002 002 002 002 002 003 001 003 002 001 001 003 002 002 002 003 003 002 005 001 002
  01 C    005 003 005 003 001 003 002 005 003 002 002 002 001 002 003 001 003 002 001 001 003 002 002 002 003 003 002 005 001 002
  01 D1   C03 XIF C03 SAE SAJ SAA PSF C03 SAA NUS N/C N/C SAF CTG C03 SAA D/K N/C SAC N/C N/C SAA N/A C01 N/C XTF N/C C03 RES IMY
  01 D2   DIS RIA SAF                 RTH CTG                     PMW SAC                             CTI     CTG     PMW CTI
  01 D3           CTI                 PMW                                                             CTH
  01 D4                               SAE
```

```
QUESTION S20            THERE IS A POOR ACCOUNTABILITY STRUCTURE IN MOST DEVELOPMENT
                        PROJECTS, LEAVING SOME QUESTION AS TO WHO IS RESPONSIBLE FOR
                        VARIOUS PROJECT FUNCTIONS.
                        A. THIS PROBLEM IS:
                           CRITICAL      (1)    AN IRRITANT          (3)
                           IMPORTANT   (2)      OF NO CONSEQUENCE   (4)
                           PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                        B. THIS IS A PROBLEM IN:
                           MANAGEMENT (1)    BOTH      (3)
                           TECHNOLOGY (2)    NEITHER   (4)
                           NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                        C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                           MANAGEMENT (1)    BOTH      (3)
                           TECHNOLOGY (2)    NEITHER   (4)
                           NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                        D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
           RESPONDOR
PART/SUB   101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A     002 004 002 003 002 002 002 001 001 002 002 002 002 001 004 002 001 001 001 004 002         001 002 001 002 001 004 002
  01 B     001 004 001 001 001 001 001 001 001 001 001 001 001 003 005 001 001 001 001 004 001         001 001 001 001 001 005 003
  01 C     001 004 001 003 001 001 001 001 001 001 001 001 001 003 005 001 001 001 001 004 001         001 001 001 001 001 005 003
  01 D1    OMA N/C OMA N/C OMA PMA OMA N/C OMA OMG OMG SAJ N/C C03 C03 OMA OMA OMA OMA OMA N/C         N/C OMA CMI N/A OMA C01 OMA
  01 D2    CMI     CMI         CMI CMI     CMI                 N/C CTF     CTF                                             N/C OMH
  01 X                                                                                         MIS MIS

           RESPONDOR
PART/SUB   201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A     002 005 002 001 003 005 004 004 001 001 002 002 002 003 003 005 003 002 001 001 002 003 004 002 002 001 002 005 002 002
  01 B     003 005 MIS 003 001 005 005 005 001 001 001 001 001 001 003 005 001 003 001 001 001 001 001 001 001 001 001 005 001 001
  01 C     003 005 MIS 003 001 005 005 005 001 001 001 001 001 001 003 005 001 003 001 001 001 001 001 001 001 001 001 005 001 001
  01 D1    N/C C03 N/C OMF OMB C03 C03 C03 OMI SAJ N/C N/C OMA OMA C03 C03 OMA N/C N/C N/C N/C CTF OME CMI N/C CMI N/C C03 OMA OMA
  01 D2        NPR             DIS DIS N/C                         PHA NPR                                                 DIS     CMI
  01 D3                                                            SAK
```

```
QUESTION 521          THERE IS MUCH CONSIDERATION IN INDUSTRY CONCERNING HOW BEST TO
                      ORGANIZE FOR THE ACCOMPLISHMENT OF A PROJECT.  (E.G., SHOULD THE
                      PROJECT BE ORGANIZED AROUND THE FUNCTION, THE PROJECT, OR UNDER
                      A NEW MATRIX SYSTEM?)
                      A. THIS PROBLEM IS:
                         CRITICAL     (1)    AN IRRITANT          (3)
                         IMPORTANT    (2)    OF NO CONSEQUENCE    (4)
                         PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                      B. THIS IS A PROBLEM IN:
                         MANAGEMENT (1)    BOTH      (3)
                         TECHNOLOGY (2)    NEITHER   (4)
                         NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                      C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                         MANAGEMENT (1)    BOTH      (3)
                         TECHNOLOGY (2)    NEITHER   (4)
                         NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                      D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
            RESPONDOR
PART/SUB    101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A      002 004 004 002 002 003 004 002 003 002 002 002 002 004 004 002 002 002 002 003 002         001 001 003 002 001 004 002
  01 B      001 004 005 001 001 001 005 001 003 001 001 003 001 005 005 001 001 001 001 001 001         001 001 001 001 001 005 001
  01 C      001 004 005 001 001 001 005 001 003 001 001 003 001 005 005 001 001 001 001 001 003         001 001 001 001 001 005 001
  01 D1     OMD OMD CO3 NOS OMZ OMZ CO3 N/C OMZ OME OME OME N/C CO3 CO3 CO3 OMZ OME OME OMZ N/C         N/C N/C UNK OMC OMZ CO1 D/K
  01 D2             OMZ             N/C                         N/C DMZ N/C                                             N/C
  01 X                                                                                      MIS MIS

            RESPONDOR
PART/SUB    201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A      002 002 002 001 001 002 002 002 002 002 002 002 004 003 001 001 004 002 002 002 003 004 004 002     002 001 002 002 002
  01 B      003 003 001 003 001 003 001 001 001 001 001 001 004 001 001 001 004 003 001 001 001 001 001 001     001 001 003 001 001
  01 C      003 003 001 003 001 003 001 001 001 001 001 001 MIS 001 001 001 004 003 001 001 001 001 001 001     001 001 003 001 001
  01 D1     N/C OMZ CO3 OMZ OMD N/C OMD CO3 OMD CPX N/C N/C N/C NOT CO3 OMD N/C OME OME N/C DMZ DMZ OME NOS     OMD N/C OMZ OMD OMD
  01 D2             OMZ                 OME N/C                         OMZ                                     OME
  01 X                                                                                                      MIS
```

```
QUESTION 322                  IT IS DIFFICULT TO IMPOSSIBLE FOR A PROJECT MANAGER TO HAVE THE
                              REQUISITE VISIBILITY, TO BE ABLE TO DETERMINE WHETHER THE PROJECT
                              IS ON SCHEDULE AND WITHIN COST.
                              A. THIS PROBLEM IS:
                                 CRITICAL     (1)   AN IRRITANT          (3)
                                 IMPORTANT  (2)     OF NO CONSEQUENCE   (4)
                                 PROBLEM INCORRECTLY OR CONFUSING (ADDED) (5)
                              B. THIS IS A PROBLEM IN:
                                 MANAGEMENT (1)    BOTH        (3)
                                 TECHNOLOGY (2)    NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                              C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                                 MANAGEMENT (1)    BOTH        (3)
                                 TECHNOLOGY (2)    NEITHER   (4)
                                 NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                              D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    004 004 002 003 002 002 004     001 001 001 002 002 001 001 002 001 002 002 004 002         001 001 001 001 001 002 001
  01 B    001 004 001 003 001 001 005     001 001 001 001 001 001 001 003 001 003 003 004 003         001 001 003 001 001 003 003
  01 C    001 004 001 003 001 001 005     001 001 001 001 001 001 MIS 003 001 003 003 004 003         001 001 003 001 001 003 003
  01 D1   C05 CMD CMD NOS CMD CMD C03     CMD CMD CMD CMD N/C N/C RTA N/C CMF CMQ CMQ CMA N/C         N/C CMM PMO N/C CMD RES O/K
  01 D2   PMA                 OMA N/C     CMO CMF CMF SAJ         CMD                                         CTA     CMF
  01 D3   OMD                 SAI             CMA CMA             CMP                                                 CMQ
  01 D4   CMK
  01 D5   CMM
  01 X                                MIS                                                         MIS MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    001 001 005 001 005 001 001 002 001 001 002 002 002 001 001 005 001 001 001 001 002 002 001 002 001 001 002 005 001 MIS
  01 B    003 003 003 003 005 003 001 001 001 001 003 001 001 001 001 005 003 003 001 003 003 001 001 003 001 001 001 005 003 MIS
  01 C    003 003 003 003 005 003 001 001 001 001 003 001 001 001 001 005 003 003 001 003 003 001 001 003 001 001 001 005 003 MIS
  01 D1   N/C NPR C03 PMK C03 SMA CMA NOS PMA CMN N/C N/C CMM CMM C01 C05 CMA N/A CMF N/C N/C CMA N/A CMF N/C CMD N/C C03 C01 CMK
  01 D2       CMD CMR XMF CMD CMD         PMM                     PMA NPR                                             NPR CMF C1B
  01 D3                                   PMN                     CMM                                                     CMD
  01 D4                                   CMD
```

```
QUESTION 525            THERE IS A GENERAL LACK OF TRACEABILITY FROM THE REQUIREMENT
                        SPECIFICATIONS TO THE FINAL CODE.
                        A. THIS PROBLEM IS:
                           CRITICAL     (1)    AN IRRITANT         (3)
                           IMPORTANT    (2)    OF NO CONSEQUENCE   (4)
                           PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) (5)
                        B. THIS IS A PROBLEM IN:
                           MANAGEMENT (1)    BOTH      (3)
                           TECHNOLOGY (2)    NEITHER   (4)
                           NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                        C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                           MANAGEMENT (1)    BOTH      (3)
                           TECHNOLOGY (2)    NEITHER   (4)
                           NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                        D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
```

	RESPONDOR																													
PART/SUB	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130
01 A	001	002	002	002	002	001	003	002	002	001	001	002	001	001	005	002	003	002	002	003	002	003		001	001	002	001	001	002	002
01 B	003	003	001	003	003	001	001	003	002	001	001	002	002	001	005	003	003	002	002	003	003	003		003	001	003	001	001	003	001
01 C	003	003	001	003	003	001	001	003	003	001	001	002	002	001	005	003	003	002	002	003	003	002		003	001	003	003	001	003	001
01 D1	CTC	XTB	CTB	NOS	XMF	CTB	CMI	N/C	RTA	CTB	CTB	CTH	N/C	N/C	C03	N/C	C02	RTI	RTI	CMH	CTB	N/C		C02	CTG	CTB	C02	NPR	C01	CTG
01 D2					RMF				CMK						CTE		SAJ				CTG			N/C			XTG	CTH	NOS	CMK
01 D3																					XIM									
01 X																							MIS							

	RESPONDOR																													
PART/SUB	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
01 A	002	002	002	001	001	004	001	001	003	002	002	002	003	002	001	005	002	001	002	001	003	002	002	004	001	002	001	002	001	003
01 B	003	003	002	003	002	005	001	003	003	001	003	002	003	003	003	005	003	003	001	003	001	002	001	004	003	003	001	003	003	003
01 C	003	003	003	003	002	005	001	003	003	001	003	002	003	003	003	005	003	003	001	003	001	003	001	004	003	003	001	003	003	003
01 D1	N/C	NPR	C03	XTF	XTM	C03	RTC	CTB	CTB	CTH	N/C	N/C	CMD	CTG	C02	C03	CTH	N/C	CTB	N/C	XTM	N/A	N/A	DIS	N/C	CTG	N/C	NPR	C01	CTH
01 D2		CTH	CTG	XMF		DIS	XTM								CTH	DIS										CTH		PMW	CTH	
01 D3		CTG	PMW												CTG															

```
QUESTION 324              THERE IS, IN GENERAL, AN INABILITY TO MEASURE THE QUALITY OF A PROGRAM
                          A. THIS PROBLEM IS:
                             CRITICAL     (1)    AN IRRITANT          (3)
                             IMPORTANT  (2)     OF NO CONSEQUENCE   (4)
                             PROBLEM INCORRECTLY WORDED OR CONFUSING (ADDED) 85)
                          B. THIS IS A PROBLEM IN:
                             MANAGEMENT (1)     BOTH        (3)
                             TECHNOLOGY (2)     NEITHER   (4)
                             NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                          C. THIS PROBLEM CAN BE SOLVED THROUGH IMPROVEMENT IN:
                             MANAGEMENT (1)     BOTH        (3)
                             TECHNOLOGY (2)     NEITHER   (4)
                             NOT A PROBLEM, UNANSWERABLE OR DON'T KNOW (ADDED) (5)
                          D. HOW WOULD (DID) YOU SOLVE THIS PROBLEM ?
          RESPONDOR
PART/SUB  101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  01 A    002 004 005 002 002 003 002 002 003 002 002 004 004 001 005 004 002 002 002 003             001 003 002 005 004 004 002
  01 B    002 005 005 003 001 003 003 003 005 003 003 002 005 001 005 005 002 003 003 003             002 003 001 005 005 005 003
  01 C    002 005 005 003 001 003 003 003 005 003 003 002 005 001 005 005 002 003 003 003             002 003 001 005 005 005 003
  01 D1   C05 C04 C03 NOS RTA CFU CID N/C C03 NOT NOT SUB C03 N/C C03 C03 CID RMY RMY CMG             N/C CME RTA C03 C03 C03 D/K
  01 D2   RTA N/C CFU         CMB         CMB             N/C     CMB CMY             CTI                         DIS CMH N/C
  01 D3   CID
  01 D4   ITH
  01 X                                                                                    MIS MIS MIS

          RESPONDOR
PART/SUB  201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
  01 A    005 002 003 002 003 002 001 004 001 003 002 001 002 002 003     002 001 002 002 003 002 002 003 002 001 002 001 002 003
  01 B    005 003 001 003 005 003 001 005 003 002 002 002 001 002 MIS     003 002 003 003 003 002 002 002 003 001 003 003 002 002
  01 C    005 003 001 003 005 003 001 005 003 002 002 002 001 002 MIS     003 002 002 003 003 002 002 002 003 001 003 003 002 002
  01 D1   C03 NOS C03 CTH C03 NOS CTI C03 CTI SUB N/C CTH RME RTA N/C     CTG N/C C03 N/C N/C CTI NOT CTI N/C CTG N/C C03 CTI N/C
  01 D2   DIS     SAT     SAJ         CID                                         N/C                 CTH                 TRU
  01 D3                                                                                                                   NUS
  01 X                                                                MIS
```

SECTION 3
REFERENCES

INTRODUCTION

The hypothesis or propositions used in this survey were formulated by the author from the literature, case studies, and his own experience. This section lists the references and case studies on which these propositions were based.

In formulating the hypothesis or propositions, each of the problems that had been recorded in the literature as being a software engineering problem was looked at from the viewpoint of a newly assigned or newly promoted project manager, working alone in a potentially unfriendly environment, and faced with all the decisions he or she must make to successfully deliver the project on time, within cost, and satisfying the user's requirements. Therefore, the statements made by the authors in the literature have been rewritten in an attempt to reflect how a project manager might view the problem, not necessarily how it was stated by the referenced author.

Following the references are the 20 propositions and the specific sources for each. To insure complete honesty of reporting, each proposition is followed by an exact quote from the reference, including page number, so that the readers can judge for themselves whether or not the author interpreted a reference correctly. Where there is no reference underneath one of the propositions, the proposition is wholly that of the author. As an aside, the number of references behind a proposition gives some weight to the seriousness and/or at least the visibility of that particular proposition.

Although the author attempted to obtain a primary source when a source document referenced another document, if the primary source wasn't readily available the secondary source was quoted.

It should be pointed out that the initial propositions and questionnaires were composed in June 1977 and distributed between August and November of that year. These same propositions were updated, clarified and rewritten in November 1977, and widely distributed in the form of a second questionnaire from November 1977 through December 1978. You may note that some of the references are dated after 1977. This was done for purely academic reasons to show that the problem has not gone away. It is of course entirely possible that the problems reported in the 1978 sources were influenced by the questionnaire and not the other way around. The author leaves this unresolved question to the reader.

REFERENCES

[ADP Org Workshop, 1978] --- Invited Workshop on "Organizing ADP Project," Cosponsored by: National Bureau of Standards, IEEE Computer Society, and Federal Interagency Committee on Automatic Data Processing (13-14 Jun 1978)

[Aron, NATO Conf, 1969] --- J. D. Aron, "Estimating Resources for Large Programming Systems," Software Engineering Techniques: Report on a Conference Sponsored by the NATO Science Committee, J.N. Buxton and B. Randell, Editors (27-31 Oct 1969), pp. 68-79

[BMDATC SOW SW-A-44-75, 1974] --- Statement of Work, "Reliable Software," SW-A-44-74, Ballistic Missile Defense Advanced Technologies Center (BMDATC), Huntsville, AL (25 Sep 1974)

[BMDATC SOW SW-A-88-75, 1974] --- Statement of Work, "Data Processing System Requirements," SW-A-88-75, Ballistic Missile Defense Advanced Technologies Center (BMDATC), Huntsville, AL (9 Dec 1974)

[Boehm, 1972] --- Barry W. Boehm, "Software and Its Impact: A Quantitative Assessment," Technical Report P-4947, The Rand Corporation, Santa Monica, CA (Dec 1972)

[Boehm, Brown and Lipow, 1976] --- Barry W. Boehm, J. R. Brown, and M. Lipow, "Quantative Evaluation of Software Quality," Proceedings, Second International Conference on Software Engineering, IEEE-CS, pp 592-605 (Oct 1976)

[Boehm, 1974] --- Barry W. Boehm, "Software Engineering," IEEE Transactions on Computers, Volume C-25, Nr 12, pp 1227-1230 (Dec 1976)

[Brooks, 1974] --- Frederick P. Brooks Jr., "The Mythical Man-Month," DATAMATION, Volume 20, Nr 12 (Dec 1974)

[CCIP-85, 1972] --- "Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980's," (CCIP-85), Volume 1: Highlights, SAMSO TR72-141 (Apr 1972)

[Cooper, 1978] --- John D. Cooper, "Corporate Level Software Management," IEEE Transactions on Software Engineering, Volume SE-4, Nr 4, pp 319-326 (Jul 1978)

[De Roze, 1976] --- Barry C. De Roze, Letter, "DOD Defense System Software Management Program," Office of the Assistant Secretary of Defense, Washington, D.C. (1 Mar 1976)

[De Roze and Nyman, 1978] --- Barry C. De Roze and Thomas H. Nyman, "The Software Life Cycle A Management and Technological Challenge in the Department of Defense," IEEE Transactions on Software Engineering, Volume SE-4, Nr 4, pp 309-318 (Jul 1978)

[Keen and Gerson, 1977] --- Peter F. W. Keen and Elihu M. Gerson, "A Politics of Software System Design," DATAMATION, Volume 23, Nr 11, pp 80-84 (Nov 1977]

[Keider 1974] --- Stephen P. Keider, "Why Projects Failed," DATAMATION, Volume 20, Nr 12, pp 53-55 (Dec 1974]

[Klass, 1978] --- Phillip J. Klass, "NORAD Data System Has 100% Overrun," Aviation Week and Space Technology, Volume109, Nr18, pp 61-63 (30 Oct 1978)

[Kolence, NATO Conf, 1968] --- K. Kolence, In a discussion on "Software Engineering Management and Methodology," Software Engineering: Report on a Conference Sponsored by the NATO Science Committee, Peter Naur and Brian Randell, Editors (7-11 Oct 1968), p 13.

[Lipow and Thayer 1977] --- M. Lipow and T. A. Thayer, "Prediction of Software Failures," Proceedings 1977 Annual Reliability and Maintainability Symposium, pp 1-6 (1977)

[Manley, 1975] --- Lt Col John H. Manley, "Embedded Computer System Software Reliability," Defense Management Journal, Volume 11, Nr 4, pp 13-18 (Oct 1975)

[McCarthy, 1975] --- Rita McCarthy, "Applying the Technique of Configuration Management to Software," Defense Management Journal, Volume 11, Nr 4, pp 23-28 (Oct 1975)

[Miller, 1975] --- Barry Miller, "Avionics Problems Bar Debute of F-15 With TAC," Aviation Week Space Technology, Volume 103, Nr 12, pp 23-25 (Dec 1975)

[Ogdin, 1972] --- Jerry L. Ogdin, "Designing Reliable Software", DATAMATION, Volume 18, Nr 7, pp 71-78 (Jul 1972)

[RADC R&D Program, 1977] --- Rome Air Development Center (RADC), Unpublished R&D Program in Software Cost Reduction (FY 1977)

[Ruth, 1974] --- Captain Stephen R. Ruth, USN, "What can the Navy learn from ALS?," Unpublished document (Approx 1974)

[Schwartz, NATO Conf, 1969] --- Jules I. Schwartz, "Analyzing Large-Scale System Development", Software Engineering Techniques: Report on a Conference Sponsored by the NATO Science Committee, J. N. Buxton, and B. Randell, Editors (27-31 Oct 1969), pp 122-137

[Slaughter, 1973] --- John B. Slaughter, "Understanding the Software Problem," Proceedings of a Symposium on the High Cost of Software, edited by Jack Goldberg, Stanford Research Institute, Menlo Park, CA, pp 41-52 (17-19 Sep 1973)

[Spier, 1976] --- Michael J. Spier, "Software Malpractices - A Distasteful Experience," Software - Practices and Experience, Volume 6, pp 293-299 (1976)

SRWG Report, 1975] --- Findings and Recommendations of the Joint Logistics Commanders Software Reliability Work Group (SRWG Report), Volume 1: Executive Summary (Nov 1975)

[Thayer, 1974] --- Richard H. Thayer, "The Rome Air Development Center R&D Program in Computer Languages and Software Engineering," RADC Tech Report Nr TR-74-80 (Apr 1974)

[Walsh, 1977] --- Dorothy A. Walsh, "Structured Testing," DATAMATION, Volume 23, Nr 7, pp 111-118 (Jul 1977)

QUESTION 5 (Problem 1) -- Performance specifications are frequently incomplete, ambiguous, inconsistent, machine dependent, and/or unmeasureable.

"People don't understand the completed operational system. . . . it is sometimes unclear that even the customer knows what he wants." [Schwartz, NATO Conf, 1969, p 129]

"The most serious problems [facing the Air Forces' command and control data processing in 1985] involve the following areas: requirements analysis and design techniques for command and control information systems and automated aids to command and control systems exercising." [CCIP-85, 1972, p 34]

Some of the principle causes of unreliable, unresponsive and incompatible software quality are, " . . . inaccurate statement of requirements by user [and] inadequate understanding of user requirements." [Slaughter, 1973, p 47]

"Research objectives: the objectives of this program are to devise and demonstrate an advanced engineering methodology that supports the design, development, validation and unambiguous communications of complete and consistent data processing subsystem performance requirements in a highly visible and traceable manner." [BMDATC SOW SW-A-88-75, 1974]

"The project is not adequately defined." "Definition of the project is vague, misleading, or totally wrong." "Project completion elements are not defined." [Keider, 1974]

"The fact that requirements often change during system development is generally acknowledged to be one of the greatest sources of cost escalation and schedule slippages in the acquisition of major software systems." [SRWG Report, 1975]

"Software . . . in embedded computer system acquisition [has] inadequate requirements analysis." [De Roze, 1976, p I-2]

"Ideally, one would like to have complete, consistent, validated, unambiguous, machine-independent specifications of software requirements before proceeding to software design." [Boehm, 1976, p 1230]

"Specific goals of this area are the development of standard tools and procedures which specify requirements. . . . Over 65% of all software errors are generated in the design phase, and most of these can be traced to poorly or inaccurately stated requirements." [RADC R&D Program, 1977, p 5]

"The requirements document was poor (weak and poorly defined, didn't satisfy the user's needs, kept changing during the course of the project, poor controls on change, etc.)." [ADP Org Workshop, 1978]

"The lack of means to produce clear, concise, and unambiguous statements of user requirements is one of the biggest contributors to the high cost of software." [De Roze and Nyman, 1978, p 310]

"First, the customer needs to be able to write complete, correct, and unambiguous specifications for the software he wants to acquire." [Cooper, 1978, p 325]

QUESTION 6 (Problem Nr 2) -- There are no decision rules for the software engineering project manager to use in selecting the correct software design techniques or tools available within the state-of-the-art.

"Programming management will continue to deserve its current poor reputation for cost and schedule effectiveness until such time as a more complete understanding of the program design process is achieved." [Kolence, NATO Conf, 1968, p 7]

"Adherence to standards and specifications is either not defined or, if defined, not followed." "Knowledge of 'tools' to perform the project more effectively is lacking." [Keider, 1974]

"The lack of application of system engineering methodology to computer system design is the root of a number of critical problems in the development of major weapons systems." [SRWG Report, 1975, p 55]

"The development of software requires a major investment in support tools and facilities. If they are not available from previous programs and not provided for in the development plans, a major schedule slippage and cost overrun can result." [SRWG Report, 1975, p 58]

"The design of software traditionally has been a craft rather than an engineering discipline. Consequently, it has tended to be unstructured, with few rules and constraints." [SRWG Report, 1975, p 59]

"Software . . . in embedded computer system acquisition [has] inconsistent application of tools and procedures." [De Roze, 1976, p 1-2]

"He needs more powerful analysis tools to help him sort out the [design] alternatives." [Boehm, 1976, p 1230]

"Tools and procedures applicable to the major technical areas of software design, development and debuggings/integration will be explored. Various existing tools and procedures will be evaluated for possible application in the Air Force. . . . Inputs from this work will provide valuable guidance for the selection and development of promising tools and procedures which will be integrated into USAF software development environment." [RADC R&D Program, 1977, p 14]

"The goal of [the software development] technical area is the development of tools and procedures which assist in the production of quality software." [RADC R&D Program, 1977, p 15]

"A particularly difficult problem is the choosing from among the variety of demonstrated alternative methods, those software tools and controls best suited for a particular program manager's needs. Thus, a key role of the S&T [Science and Technology] program will be to survey the successful software methods, and place them into perspective from the program manager's viewpoint." [De Roze and Nyman, 1978, p 313]

QUESTION 7 (Problem 3) -- There is no measure or index of "goodness" of code that can be used as an element of software design, and there is no practical way to guarantee one program is better than another.

"We can look at two programs, ostensibly similar, and find that they accomplish the same desired function. Yet, we will say that this program is a good program, while that one is a bad program, but we don't always know why." [Ogdin, 1972, p 71]

Some of the principle causes of unreliable and unresponsive software quality are " . . . lack of standards by which performance can be measured [and] poor testing and certification practices." [Slaughter, 1973, p 47]

"We recognize nowadays that the more programs we write for applications of increasing complexity and sophistication, the more uncertain we become of our ability to write these programs such that they be as 'good' as possible." [Spier, 1976, p 293]

QUESTION 8 (Problem 4) -- There are no decision rules for selecting the procedures, strategies, and tools to be used in testing software.

"The contractor shall identify, define, and recommend test tools and techniques which are candidates for application to each element of the software engineering approach as determined in the Software Engineering Program [being pursued by BMDATC]." [BMDATC SOW SW-A-44-75, 1974]

"The lack of application of systems engineering methods to the design of software has led to systems that are non-modular, lacking well established interfaces, and difficult to test." [SRWG Report, 1975]

"Software . . . in embedded computer system acquisition [has] inconsistent application of tools and procedures." [De Roze, 1976, p I-2]

"Software quality specifications and trade-offs [reflect a problem in] lack of quantitative test standards." [De Roze, 1976, p I-12]

"There has been no systematic attempt to develop complementary verification techniques geared to determine software reliability as software develops. . . . Testing continues to be 'a witch hunt'." [Walsh, 1977]

"This [software debug and/integration] technical area is concerned with the development of tools that support the software debugging integration activities." [RADC R&D Program, 1977, p 15]

"The goal of this [testing] technical objective is to achieve more comprehensive tests per dollar expended for Air Force software." [RADC R&D Program, 1977, p 19]

QUESTION 9 (Problem 5) -- There is no measurement, or index of reliability that can become an element of design and there is no way to predict software failure; i.e., there is no practical way to guarantee (prove) the delivered software meets a given reliability criteria.

" . . . There is no compreshensive and formalized set of techniques which will . . . support design in quality or reliability across the development cycle." " . . . The contractor shall identify and define the heuristics related to design practices and development tools which impose the "designed in" concept of reliable software for each element of the Software Engineering Program." "The contractor should define a procedure for combining the application of the test tools identified . . . into a comprehensive reliability measurement technique." [BMDATC SOW SW-A-44-75, 1974]

"Because of the varied nature and sources of these errors, it has been extremely difficult to satisfactorily define or measure computer software reliability." [Manley, 1975, p 14]

"The final product was often characterized by lack of reliability, which refers to the ability of a program to produce correct results when given a specific input." [Manley, 1975, p 24]

"Since the field of software reliability engineering is new, it lags behind the hardware reliability discipline and available methods, tools and techniques." "Software reliability management techniques are virtually non-existent, and therefore must be evolved." [SRWG Report, 1975]

"Problems in software quality assurance and control [show a] lack of management monitoring of software reliability." [De Roze, 1976, p I-5]

"Development of adequate analytical models and tools for predicting cost, reliability and maintainability in software [are required]." [RADC R&D Program, 1977, p 6]

"The goal of [the software data repository] is to create a centralized repository that will serve as a focal point for exchange and analysis of software data collected from . . . software projects, personnel, companies and across different programming languages, and will provide an invaluable tool in establishing standards of performance, measures of software reliability and quality." [RADC R&D Program, 1977, pp 19-20]

"It is desirable to be able to predict at the earliest possible moment the resources needed to correct the causes of software failure and determine where within the software to devote the most resources." [Lipow and Thayer, 1977]

"A good software contract needs to pertain to reliability and maintainability requirements. Since these concepts have not yet been defined, how can they be included in a legal document? Assume for the moment that the foregoing could be done, how then could the software be evaluated to determine if it were in conformance with the correct specification?" [Cooper, 1978, p 325]

QUESTION 10 (Problem 6) -- There is no way to guarantee that the delivered software meets the user's requirements.

"One of the . . . most serious problems [facing command and control data processing in 1985] involved. . . techniques for guaranteeing that command control information systems have no software 'bugs' which could cause a system to malfunction at critical times." [CCIP-85, 1972, p 34]

QUESTION 11 (Problem 7) -- There is no measurement or index of maintainability that can become an element of software design; i.e., there is no practical way to guarantee that a given program is more maintainable than another.

"Development of adequate analytical model and tools for predicting cost, reliability and maintainability in software [are required]." [RADC R&D Program, 1977, p 6]

"One [research] effort that is currently underway will develop techniques for the construction of self-checking software in the evaluation of computer program designs in terms of a quantitative measure of maintainabilty." [RADC R&D Program, 1977, p 22]

"A good software contract needs to pertain to reliability and maintainability requirements. Since these concepts have not yet been defined, how can they be included in a legal document? Assume for the moment that the foregoing could be done, how then could the software be evaluated to determine if it were in conformance with the correct specification? [Cooper, 1978, p 325]

QUESTION 12 (Problem 8) -- No technical discipline exists for the design of maintainable programs.

"The integration of operational support requirements and the transition from production into operational use are high on the list of major problems and weapons systems acquisition. The lack of transferability of software, the lack of provisions for the maintenance, and the cost of changes resulting from these inadequacies have been cited in many previous software studies as important problems needing solutions." [SRWG Report, 1975]

"Software maintenance is an extremely important, but highly neglected activity." [Boehm, 1976, p 1235]

"Development of adequate analytical models and tools for predicting cost, reliability, and maintainability of software [are required]." [RADC R&D Program, 1977, p 6]

"In the future the DOD will need to take a comprehensive look at the life-cycle approach to acquiring software. This will include the formulation of design and management principles, development and validation of software life-cycle cost models, and evaluation of design methodologies for ease of maintenance and program update." [De Roze and Nyman, 1978, p 311]

QUESTION 13 (Problem 9) -- Planning for software engineering projects is generally poor.

" . . . the CCIP-85 study found that the problems of software productivity on medium or large projects are largely the problems of management: of thorough organization, good contingency planning, thoughtful establishment of measureable project milestones, continuous monitoring on whether the milestones are properly passed, and prompt investigation and corrective action in case they are not." [Boehm, 1972, p 16]

"Little or no time is spent in planning the project. Rather, analysis design and/or coding is begun immediately upon the project approval. The project leader is not permitted the 'luxury' of planning." [Keider, 1974]

"In general, poor planning can be blamed for most of the software industry's inability to cope with these new demands and rapid growth. Projects were initiated without a clear goal; thus, as programmers coded, they made their own assumptions of the purpose of the programs and this often adversely affected the final product. Even when the goals were clearly specified the development process suffered from inappropriate planning." [McCarthy, 1975, p 23]

"In order to insure that development of major software subsystems is well organized and managed, and all requirements are properly understood and defined, it is essential to have a detailed development plan prepared and evaluated prior to starting full scale development." [SRWG Report, 1975]

"Although the program manager should have on his immediate staff system engineers who are knowledgeable about software, manpower limitations often restrict staff to a skeleton organization. Without other direct support, the program manager cannot adequately fulfill his responsibilities for carrying out the extensive planning and monitoring associated with the major new weapon system." [SRWG Report, 1975]

"To some extent an Air Force official conceded,'we underestimated the size of the task'." [Miller, 1975, p 23]

"Lack of planning and operation guidance in day-to-day operation is a problem facing management." [De Roze, 1976, p I-8]

There is . . . "poor planning: generally this leads to large amounts of wasted effort and idle time because of tasks being unnecessarily performed, overdone, poorly synchronized, or poorly interfaced." [Boehm, 1976, p 1237]

"The thrust of this area [planning] is to develop and validate models of the software production process . . . [which] enable the generation, evaluation and refinement of uniform procedures for setting goals, estimating resources expenditures (time, money, material and manpower), and for revising plans efficiently and effectively based on internal or external world events . . ." [RADC R&D Program, 1977, p 9]

"Not enough time was spent doing technical planning, defining the scope of work, and developing the technical tasks list." [ADP Org Workshop, 1978]

QUESTION 14 (PROBLEM 10) -- There is an inability to accurately estimate delivery time of a computer program.

"It is very common for the cost and schedule of large program systems to exceed the initial estimates . . . . Not only does the main resource--manpower--vary widely in productivity and quality, but the secondary resources, such as machine time and publications support, are frequently unavailable at the appropriate times." [Aron, NATO Conf, 1969, p 68]

"One of the . . . most serious problems [facing command and control data processing in 1985] involves . . . techniques for effecting timely development of software without compromising flexibility or functional requirements." [CCIP-85, 1972, p 34]

Problems associated with delivery schedule are caused by "poor estimation practices, variable programmer skills and productivity, unrealistic milestones." [Slaughter, 1973, p 49]

"RADC requires a good technique of estimating accurately the length of time (and consequently cost) necessary to produce a large software package." [Thayer, 1974]

" . . . our techniques of estimating are poorly developed." [Brooks, 1974]

"No standards exist for estimating how long the project will take. That is, each project is treated as a new and valuable system with some individual responsible for estimation." "Estimation is not done by the probable project leader, but rather, by whoever happens to be available at the estimating time." "Short lead times are allowed for estimates, with corresponding inaccuracies as a result." [Keider, 1974]

"Software . . . in an embedded computer system acquisition [frequently has] inaccurate cost/schedule projections." [De Roze, 1976, p I-2]

"There is poor resources estimation: without a firm idea of how much time and effort a task should take, the manager is in a poor position to exercise control." [Boehm, 1976, p 1237]

"Tools and procedures are needed by management to accurately plan/forecast the resources (time, money, manpower and facilities) required to develop a system to meet functional operational requirements." [RADC R&D Program, 1977, p 5]

"Deadlines and milestones were unrealistic and not well planned." [ADP Org Workshop, 1978]

"U.S. Air Force Program to modernize the North American Air Defense Command's Real Time Data Processing System has an overrun of more than 100% in both time and cost and will provide no better inital capability than the system it replaces." [Klass, 1978, p 61]

QUESTION 15 (Problem 11) -- The ability to plan for resources, particularly the number of programmers required, is poor.

"It is very common for the cost and schedule of large program systems to exceed the initial estimates. . . . Not only does the main resource--manpower--vary widely in productivity and quality, but the secondary resources, such as machine time and publications support, are frequently unavailable at the appropriate times." [Aron, NATO Conf, 1969, p 68]

"Problems associated with difficulty in determining software development costs are poor estimation of production cost, inadequate programmer skill levels, uncertainty of cost allocations." [Slaughter, 1973, p 48]

". . . our techniques of estimating are poorly developed." [Brooks, 1974]

"Resource requirements are not scheduled for the project. Critical items, such a keypunch, test time, user manual typing, secretarial, and printing requirements, and are addressed only after they have affected the project." [Keider, 1974]

"The lesson: It is vital to make appropriate allowances for elapsed time in total manhours to reflect the real meaning of an acceptable system. Further, be prepared to accept a drastic increase in elapsed time when, in order to compensate for delays, a significant increase in total manyears are used as a cure." [Ruth, 1974]

"Software . . . in an embedded computer system acquisition [frequently has] inaccurate cost/schedule projections." [De Roze, 1976, p I-2]

"There is poor resources estimation: without a firm idea of how much time and effort a task should take, the manager is in a poor position to exercise control." [Boehm, 1976, p 1237]

"Development of adequate analytical models and tools for predicting cost, reliability and maintainability in software [are required]." [RADC R&D Program, 1977, p 6]

"Tools and procedures are needed by management to accurately plan/forecast the resources (time, money, manpower and facilities) required to develop a system to meet functional operational requirements." [RADC R&D Program, 1977, p 5]

"Studies by industry and DOD have concluded that today there are no simple universal rules for predicting software costs accurately, and that to do so required an understanding of the nature of the individual program and the individual routines within that program." [De Roze and Nyman, 1978, p 310]

"U.S. Air Force Program to modernize the North American Air Defense Command's Real Time Data Processing System has an overrun of more than 100% in both time and cost and will provide no better initial capability than the system it replaces." [Klass, 1978, p 61]

QUESTION 16 (Problem 12) -- There is no real quality method of designing a project control plan that will enable project managers to control their project.

"The main interest of management is in knowing what progress has been made towards reaching the final goal of the program. The difficulty is to identify observable events which mark this progress." [Kolence, NATO Conf, 1968, p 56]

"Because project milestones are not determined at the onset of a project, percentage of completion is usually equated to percent of hours expended." [Keider, 1974]

"There is poor resources estimation: without a firm idea of how much time and effort a task should take, the manager is in a poor position to exercise control." [Boehm, 1976, p 1237]

"The control area is aimed at three interrelated areas; tracking resource expenditures, controlling deliverables and changes (configuration management), and improving the ability to assess both cost/schedule performance as well as technical performance during development in terms of activity completion in end products sufficiency." [RADC R&D Program, 1977, p 9]

"The program manager's choice of effective software management and control techniques is further complicated by lack of measurement criteria and experience data." [De Roze and Nyman, 1978, p 314]

"In spite if this position, corporate management has few direct control mechanisms for enforcing policy and standards among projects." [Cooper, 1978, p 324]

QUESTION 17 (Problem 13) -- There are no decision rules for the selection of management techniques for software engineering project management.

"The software acquisition management standards [reflect a] lack of consistent policy and planning guidance (via standards, regulations, instructions)." "The software acquisition management standards [reflect a] lack of standard terminology governing software acquisition and management." [De Roze, 1976, p I-7]

QUESTION 18 (Problem 14) -- Techniques for the selection of project managers are poor, generally resulting in poorly managed projects.

"Change of personnel is one of the major reasons why projects fail. Personnel, including project leaders, are removed from the project, with no adjustment to the schedule for the time lost due to the change." [Keider, 1974]

"A wide variation exists in the degree in which program managers are staffed with personnel competent in system engineering and software applications." [SRWG Report, 1975]

"Personnel development and training [problems show a] shortage of practitioners." [De Roze, 1976, p I-10]

" . . . as a very general statement, software personnel tend to respond to problem situations as designer rather than as managers." [Boehm, 1976, p 1237]

"Problem/issue . . . insufficient understanding by managers . . . [in the] acquisition of software." [De Roze, 1976, p I-8]

"The project manager position was weak (was inexperienced, didn't assume leadership, didn't communicate with the user, etc.)." [ADP Org Workshop, 1978]

QUESTION 19 (Problem 15) -- There is no means of measuring with any degree of accuracy the quality and quantity of code expected from a programmer.

"There is a great deal of difference in programmer capability . . . [which] does make it much more difficult to schedule and to measure progress in the early phases." [Schwartz, NATO Conf, 1969, p 130]

"It is very common for the cost and schedule of large program systems to exceed the initial estimates . . . Not only does the main resource--manpower--vary widely in productivity and quality, but the secondary resources, such as machine time and publications support, are frequently unavailable at the appropriate times." [Aron, NATO Conf, 1969, p 68]

Problems associated with delivery schedule are caused by " . . . poor estimation practices, variable programmer skills and productivity, unrealistic milestones." [Slaughter, 1973, p 49]

"RADC is interested in any behavioral studies on programmers, their profiles, qualifications of good programmers, and any other data which will lead to better selection procedures, training, etc." [Thayer, 1974]

"The [research] goal of [the software data repositor] is to create a centralized repository that will serve as a focal point for exchange and analysis of software data collected from . . . software projects, personnel, companies and across different programming languages, and will provide an invaluable tool in establishing standards of performance, measures of software reliability and quality." [RADC R&D Program, 1977, pp 19-20]

"The best practitioners, individual superstars, can produce high quality and efficient software on schedule at low cost, however, it is not very useful to decree an approach to 'hire just good programmers'. In fact, there is no scientific measure of either quality of software or the performance of practitioners." [De Roze and Nyman, 1978, p 311]

"Many projects are pursued by the project manager completely unaware that skills available to him are within his own shop. A skills inventory of past accomplishments of each staff member simplifies the staffing of a project and assures the experience is 'recyclable'." "Staff or members are considered 'universal experts'. During estimation stage, and again during implementation, staff members are considered to be equally competent analysts, designers, programmers, librarians, documentation specialists, etc. They are assigned to any of these functions with little consideration given to their abilities." "Personnel are not evaluated. There is an ideal time and only one, to evaluate the performance of an individual on a project and that is immediately at the conclusion of a project. Yet, only too often, personnel evaluation is tied to prominent anniversary dates." [Keider, 1974]

"It is a well known (and documented) maxim that productivity of [analysts, system programmers and applications programmers] is little understood, and, as a result, seldom evaluated. As a result, two persons, each earning $25,000, could be producing at a rate of X and 5X and very little is typically done about the disparity." "These issues combine with the turnover rate which reflects on the man with the 5X as well as the X producer [and] leads to the lesson: Never assume stability of data processing skills area for the life of the project. Make allowances for significant turnovers and assume that the high producer will go first, requiring far more than a 1 for 1 relief." [Ruth, 1974]

QUESTION 20 (Problem 16) -- There is a poor accountability in most development projects, leaving some question as to who is responsible for various project functions.

"Project leader responsibility is undefined." "It sounds strange, but many projects flounder through to completion without a rudder." "In general, very few installations have one man accountable for an entire project, but rather fragment the responsibility to the point where no one person is accountable." [Keider, 1974]

"Lesson: For a major project, allow senior managers an opportunity to be associated with risk of invention as well as those of limitations.

This means that for a major data systems project an officer or civilian should be appointed with the understanding that the tenure may be as long as 5 or 6 years." [Ruth, 1974]

"There is a poor accountability structure: projects are generally organized and run with very diffuse delineation of responsibilities . . ." [Boehm, 1976, p 1237]

"There wasn't a single, recognized authority in charge, coordinating staff activities." [ADP Org Workshop, 1978]

<u>QUESTION 21 (Problem 17)</u> -- There is much consternation in industry concerning how best to organize for the accomplishment of a project (e.g., should the project be organized around the function, the project, or under a new matrix system?).

(No References)

<u>QUESTION 22 (Problem 18)</u> -- It is difficult to impossible for a project manager to have the requisite visibility to be able to determine whether the project is on schedule and within cost.

"The main interest of management is in knowing what progress has been made towards reaching the final goal of the program. The difficulty is to identify observable events which mark this progress." [Kolence, NATO Conf, 1968, p 56]

"Good programs are creative . . . the art of programming requires creativity. There are non-programmers involved in most aspects of the system development process. . . . Non-programming managers usually have the option of trusting the programmer in situations or trying to inflict their judgement on the programming personnel. Either course without adequate understanding is dangerous. It is extremely difficult to judge the actual status of the system at any given time. There is little methodology used in testing or system development. No one knows the whole system." [Schwartz, NATO Conf, 1969, p 130]

" . . . our estimating techniques fallicously confused effort for progress, hiding the assumption that men and months are interchangeable." . . . schedule progress is poorly monitored." [Brooks, 1974]

"Because project milestones are not determined at the onset of a project, percentage of completion is usually equated to percent of hours expended." "The project team's activities are not clearly represented to the end user." "Posting or reporting of project information is not performed, resulting in the project leader being unaware of what completion percentage is, and the user being unaware of the impact of changes upon the original system." "Project reviews are typically exercises in trivia." [Keider, 1974]

"First, [lesson] aggressive management of data systems demands that assumptions about the hardware, applications and systems software be reevaluated frequently. Simple restatement of previous knowledge or 'gut feels' may be enough in some kinds of projects but not where the stakes are dramatically high as in a mammouth data system effort." "The second lesson has to do with bench marks. It is simply this; about every 6 months the original bench marks should be reviewed. To do this otherwise would be comparable to a navigator using dead reckoning from port to port without ever getting a firm fix on an intermediate position." [Ruth, 1974]

"The project was killed, not because it was technically unfeasible, but through the sheer frustration of management of not knowing when the project would be delivered or how much it would cost." [Source: unidentified member of a project to deliver a large real time range safety system at Vandenberg AFB, CA 1974]

"The lack of software visibility, as compared to hardware, in the acquisition of major subsystems is generally agreed to contribute to the fact that it is not well managed." "The abstract nature of software makes it difficult to measure progress, and makes it even more necessary to formalize the steps in design, and limitations, and test." [SRWG Report, 1975, p 51-53]

"There is poor control: even a good plan is useless when it is not kept up-to-date and used to manage the project." [Boehm, 1976, p 1237]

QUESTION 23 (Problem 19) -- There is a general lack of traceability from the requirements specification to the final code.

"Research objective: the objective of this program is to devise and demonstrate an advanced engineering methodology that supports the design, development, validation and unambiguous communications of complete and consistent data processing subsystem performance requirements in a highly visible and traceable manner." [BMDATC SOW SW-A-88-75, 1974]

"Most automated aids to software design provide little support for such management needs as configuration management, traceability to code or requirements, and resource estimation and control." [Boehm, 1976, p 1238]

"Specific goals of this area are the development of standard tools and procedures . . . to allow verification and tracking throughout the acquisition cycle." [RADC R&D Program, 1977, p 5]

QUESTION 24 (Problem 20) -- There is, in general, an inability to measure the quality of a program.

One of the principle causes of problems in software quality is "lack of appropriate management attention and control." [Slaughter, 1973, p 48]

" . . . there is no comprehensive formalized set of techniques which will . . . support design in quality or reliability across the development cycle." [BMDATC SOW SW-A-44-75, 1974]

"Quality control: typically, when a project is completed, it is never evaluated for quality. The quality control criteria is 'does the program run?'." [Keider, 1974]

"In general, poor planning can be blamed for most of the software industry's inability to cope with these new demands and rapid growth. Projects were initiated without a clear goal; thus, as programmers coded, they made their own assumptions of the purpose of the programs and this often adversely affected the final product. Even when the goals were clearly specified the development process suffered from inappropriate planning." [McCarthy, 1975, p 23]

"Software quality specifications and trade-off [reflect problems] in lack of quantitative quality, reliability goals and objectives." [De Roze, 1976, p I-2]

"Software . . . in embedded computer system acquisitions [has] low quality." [De Roze, 1976, p I-2]

"Why evaluate software quality? Suppose you receive a software product which is delivered on time, within budget, and correctly and efficiently performs all specific functions? Does it follow that you will be happy with it? . . . Here are some of the common problems you may find: The software product may be difficult to use or easy to misuse. The software product may be unnecessarily machine-dependent, or hard to integrate with other programs." [Boehm, Brown, and Lipow, 1976]

"There is inappropriate success criteria: minimizing development cost and schedule would generally yield a hard-to-maintain product. Emphasizing 'percent coded' tends to get people coding early and to neglect such key activities as requirements and design validation, test planning, and draft users documentation." [Boehm, 1976, p 1237]

"Another on-going [research] effort will begin to develop a working definition of software quality. The software quality definition will be built by determining the factors relating to software quality, identifying pertinent criteria that make up each factor and establishing metrics for these factors so that the objective analysis of software quality can be permitted." "RADC is developing a number of metrics for use in the quantative measurement of software." [RADC R&D Program, 1977, p 22]

"When criteria for success are ambiguous and lack a simple measure, decisions concerning system extensions will appear arbitrary from the perspective of some of the interested parties." "The less clear the definition of success and completion, the more political the decision making will be. [Keen and Gerson, 1977, p 83]

"A common pitfall is that project managers tend to be development oriented. The most pressing responsibility is the development of their system within budget and on schedule. Consequently, they optimize the development process, often at the expense of overall life-cycle cost considerations." [Cooper, 1978, p 320]

"A good software contract needs to pertain to reliability and maintainability requirements. Since these concepts have not yet been defined, how can they be included in a legal document? Assume for the moment that the foregoing could be done, how then could the software be evaluated to determine if it were in conformance with the correct specification?" [Cooper, 1978, p 325]

"In the future the DOD will need to take a comprehensive look at the life-cycle approach to acquiring software. This will include the formulation of design and management principles, development and validation of software life-cycle cost models, and evaluation of design methodologies for ease of maintenance and program update." [De Roze and Nyman, 1978, p 311]

APPENDIX A

CONTRIBUTORS

INTRODUCTION

This appendix lists those individuals (usually project managers) and firms who completed the survey. This list is provided to: (1) acknowledge the contribution, hard work, and willingness to contribute to the general knowledge of computer science by these individuals, and (2) to lend credibility to this report by making visible the excellent source of the data.

These people and companies are all members and supporters of the AIAA Technical Committee on Computer Systems.

At the end of this list is a group of individuals that wished to remain anonymous in order that they could provide more candid, truthful answers.

It was obvious from the answers received that the contributors worked very hard making the answers as truthful as possible. Again, the authors thank you.

CONTRIBUTORS

Mr. Philip S. Babel Technical Advisor for Computer Systems Acquisition	Simulator Systems Program Office Aeronautical Systems Division Wright-Patterson AFB, OH 45433
Mr. Francis J. Barrett Chief, PEACE SIGMA Development Unit	Data Automation Branch Sacramento Air Logistics Center McClellan AFB, CA 95652
Mr. Frank L. Bernstein Vice President Federal Systems Division	CALCULON Corporation 1501 Wilson Boulevard Arlington, VA 22209
Mr. Herman S. Binder Section Head, Systems Design Analysis & Integration Section	Grumman Aerospace Corporation Bethpage, NY 11714

Mr. M. Lenard Birns Program Manager, Naval Warfare Gaming System	Defense Systems Division Computer Sciences Corporation 304 West Route 38, Box N Moorestown, NJ 08057
Mr. Jack E. Bloodworth Manager, ALCM Software	The Boeing Aerospace Company P. O. Box 3999 Seattle, WA 98124
Mr. David A. Brown Chief, ARRCS Development Group	Data Automation Branch Sacramento Air Logistics Center McClellan AFB, CA 95652
Mr. Allen G. Burgess Manager, Computer Systems Laboratory	Equipment Division Raytheon Company 528 Boston Post Road Sudbury, MA 01776
Mr. George R. Cannon, Jr. Manager of Vandenberg Programs	Logicon, Incorporated P. O. Box 1567 Vandenberg, CA 93437
Mr. Frank J. Cerulli Director of Engineering Computer Systems Division also Products Systems Division	Lockheed Electronics Company, Incorporated U.S. Highway 22 Plainfield, NJ 07061
Mr. James P. Chilton Director, Data Processing Sub Systems Systems Technology Program	McDonnel Douglas Astronautics Company 5301 Bolsa Avenue Huntington Beach, CA 92647
Mr. Arthur C. Ciccolo Associate Division Leader Computer Science Division	The Charles Stark Draper Laboratories, Incorporated 555 Technology Square Cambridge, MA 02139
Mr. James W. Clark Manager of Engineering Operations	United Technologies Research Center East Hartford, CT 06108
Mr. Jerry E. Cummings Program Analyst Logistics Research & Systems Division	Directorate of Plans & Programs Sacramento Air Logistics Center McClellan AFB, CA 95652

Mr. G. Russell Curtis	General Electric Company
Manager, Simulation & Data	450 Persian Drive
Systems	Sunnyvale, CA 94086
Information Systems Programs	
Mr. Alan J. Deerfield	Submarine Signal Division
Consulting Scientist	Raytheon Company
	P. O. Box 360
	Portsmouth, RI 02871
Mr. Edward M. Dunaye	Planning Research Corporation
Director, Quality Assurance	7600 Old Springhouse Road
	McLean, VA 22101
Mr. Joe N. Dyer	Lockheed Missile & Space Company,
Manager, Equipment Evaluation	Incorporated
& Systems Programming	P. O. Box 504
	Sunnyvale, CA 94088
Mr. Richard R. Erkeneff	McDonnell Douglas Astronautics
Chief Design Engineer,	Company
Data Control & Processing	5301 Bolsa Avenue
Systems	Huntington Beach, CA 92647
Mr. S. G. Evetts	Vought Corporation
Project Manager	P. O. Box 5907
	Dallas, TX 75222
Dr. George R. Fath	General Electric Company
Acting Manager	901 Broad Street
Avionics Development	Utica, NY 13503
Engineering	
Mr. Herb Finnie	Lockheed Missile & Space Company,
Manager, PLSS Software	Incorporated
Development	P. O. Box 504
	Sunnyvale, CA 94088
Mr. J. I. Freeman	Vought Corporation
Avionics Project	P. O. Box 5907
Engineering	Dallas, TX 75222
Dr. Virgil "Smokey" V. Griffith	McDonnell Aircraft Company
Chief, Electronics Engineer	P. O. Box 416
Digital Computer & Software	St. Louis, MO 63166
Engineering	

Mr. Harvey I. Gold Manager, Software Technology Department	System Development Corporation 2400 Colorado Santa Monica, CA 90406
Dr. Kenneth A. Hales Manager, MSP Mission Control & Software	The Boeing Aerospace Company P. O. Box 3999 Seattle, WA 98124
Mr. Uwe W. Ibs Design Specialist	Pomona Division General Dynamics Corporation P. O. Box 2507 Pomona, CA 91766
Dr. Peter R. Kurzhals Director, Guidance, Control & Information Systems Division	Headquarters National Aeronautics & Space Administration Washington, DC 20546
Mr. John C. Lemanczyk Manager, Software Technology Development	Grumman Aerospace Corporation Bethpage, NY 11714
Mr. Myron Lipow Senior Staff Engineer, Product Assurance Systems Engineering & Integration Division	Defense & Space Systems Group of TRW, Incorporated One Space Park Redondo Beach, CA 90278
Mr. Austin Maher Manager, Software	Kearfoot Division The Singer Company Little Falls, NJ 07424
Dr. John H. Manley Assistant to the Director	Applied Physics Laboratory The Johns Hopkins University Johns Hopkins Road Laurel, MD 20810
Dr. Robert R. McCready Applied Mathematician	Vought Corporation P. O. Box 5907 Dallas, TX 75222
Mr. H. Lewis Parker Manager, Mini/Micro Based Systems Department	COMSTAT Laboratories 22300 Comstat Drive Clarksburg, MD 20734
Dr. Leon Pressor President	Softool Corporation 340 S. Kellogg Avenue Goleta, CA 93017

Dr. Terry A. Straeter Head, Programming Technologies Branch	Langley Research Center National Aeronautics & Space Administration Hampton, VA 23665
Mr. Herbert D. Strong, Jr. Manager, ADP Management Office Flight Projects Support Office	Jet Propulsion Laboratory California Institute of Technology 4800 Oak Grove Drive Pasadena, CA 91103
Mr. R. L. Van Tilburg Senior Scientist Computer Programming Laboratory	Hughes Aircraft Company P. O. Box 3360 Fullerton, CA 92634
Mr. Gene F. Walters Manager, Software Technologies Information Systems Program	General Electric Company 450 Persian Drive Sunnyvale, CA 94086
Mr. Lynn S. Wilson Director, West Coast Operations	Grumman Data Systems Corporation 16133 Ventura Blvd., Sutie 675 Encino, CA 91436
Mr. Eric W. Wolf Manager, Washington Operations	Bolt Beranek & Newman, Incorporated 1701 No. Fort Myer Drive Arlington, VA 22209
Anonymous Techniccal Advisor for Computers	Engineering & Development Organization Large Government Agency (Military)
Anonymous Manager, Communication Analysis	Electronic Systems Large Manufacturing Company
Anonymous Chief, Scientific Applications Analysis Branch	Research Center Large Government Agency (Non-Military)
Anonymous Tech Director, Simulation Division	Software and Engineering Large Manufacturing Company
Anonymous Senior Engineering Specialist Avionics Software	Aircraft Development Large Aerospace Corporation

Appendix B

QUESTIONNAIRE

INTRODUCTION

This appendix contains Questions 5 through 24 of Part Three, plus identification data obtained from Parts One and Two of the original questionnaire. This added information is used to provide a brief background of both the position and attributes of the participant and the type and attributes of the participant's company and is reported artificially as answers to Pseudo Questions 1, 2, 3, and 4.

The questionnaire (Questions 5 through 24) is a slightly modified version of the original questionnaire.

The author did not do a complete job of selecting possible answers for each question/problem. In particular, for part "a," there was no possible answer for those surveyees who felt that the proposition was either wrong, incorrectly stated and/or chose not to answer the question. The surveyees normally wrote their disagreement in the remarks section or in the margin of the report. This was handled by creating a fifth possible answer to part "a" entitled "problem incorrectly worded or confusing." This then created an inaccuracy in parts "b" and "c." since there was no means which a surveyee could indicate that it was not a problem, it was unanswerable because he/she did not agree with the problem, or did not know the answer. Many of the surveyees therefore either chose to ignore question "b" and "c," or wrote in their disagreement in the margin or some other appropriate place. For these reasons a possible answer of "not a problem, unanswerable, or don't know" was added.

In order to save space the possible answers were only provided one time as answers to Question 5. On the original questionnaire the same set of possible answers was repeated 19 more times. In addition, the original questionnaire had Questions 1, 2, 3, 4, and 25 which were general type questions for the participants, but did not pertain to the major issues and/or problems of software engineering project management. These original questions 1, 2, 3, 4, and 25 have been reported as part of Volume II where the author felt this was more appropriate.

In addition, some of the questions were poorly written. This is not only the opinion of the author (in hindsight) but includes the opinion of the surveyees. Therefore, based on answers to the original questionnaire, 19 of the questions were rewritten. This was done for the benefit of future researchers who might wish to make use of some or all of the questions.

Answers to part "d" of the questionnaire were narrative in nature. In order to report these answers the author included them by grouping like responses under a single code and reporting this single code on the tabulation sheet. If the reader wishes to get to the full flavor of the responses, he/she is referred to Appendix D.

The author hopes the above explanation does not appear to be too complex. This was done purely in the interest of conveying the maximum amount of information to the reader as to what the original questions were and the answers that they engendered.

A SURVEY OF MANAGEMENT TECHNIQUES AND PROCEDURES EMPLOYED IN SOFTWARE DEVELOPMENT PROJECTS

PART THREE (Modified)

Major Problems of Software Engineering Project Management

This portion of the questionnaire presents 20 propositions concerning major problems of software engineering project management. The surveyee is asked to identify himself, his company and then give his opinion on the problem, whether or not it was a management or technology problem, and whether or not it can be solved through improvements in management or technology.

THE IDENTIFICATION NUMBER ASSIGNED THIS FORM IS_______________.

Please return completed form in envelope provided or mail to:

Colonel Richard H. Thayer
SM-ALC/ACD
McClellan AFB, CA 95652

SECTION 1 - SURVEYEE IDENTIFICATION (ADDED)

1. What is your position within your company (including university, Government, etc.) and/or the computing industry (including consultants, students, etc)?

a. Line manager generally over software development (including president, VP, directors, software development division managers, etc.).

b. Project manager responsible for the successful delivery of projects (including project manager of software subsystems).

c. Individual developer generally connected with a project (including technical director, designers, analysts, engineers, programmers, etc.).

d. Senior staff position responsible for establishing broad software policy for the organization (e.g. senior technical director).

e. Supervisory/Senior staff position, software function (including miscellaneous software development, software R&D, software technology, software IV&V, software evaluation, etc.).

1. Consultant.

z. Other/Comment:____

2. Which of the following attributes apply to the surveyee?

a. R&D orientated.

b. Educator (all).

e. Programmer and/or software analysts.

f. Engineer and/or functional analysts.

g. Quality assurance/technical director

h. Manager/Supervisor

i. Government employee.

j. PhD.

k. Consultant.

l. Establishes /influences broad policy on software development.

m. National author and/or speaker on software development.

n. Affiliated with IEEE - CS, ACM, or other data processing progessional organizations.

o. Affiliated with AIAA or other aerospace professional organizations.

r. American-Canadian influence.

3. The affiliate, employer or firm of the surveyee is primarily engaged in:

b. A manufacturer of other than computer hardware

c. A "software house"

d. An engineering service and technical support organization

e. The government: federal (non-military), federal (military), state, county, municipal

f. A university, R&D laboratory, educational institution

m. The utilities: communications, electric, gas

4. Which of the following attributes apply to the surveyee's affiliate, employer, or firm?

The gross revenues (or budget) for last year were:

a. Less than 50 million dollars.

b. Between 50 million and 500 million dollars.

c. In excess of 500 million dollars.

What percent of revenue is derived from or budget alloted to software development?

d. Very little (less than 10%).

e. Moderate to average (10 to 50%).

f. High (50 to 90%).

g. Almost all (greater than 90%).

How many people:

j. Are employed by firm.______

k. Work in all aspects of software.______

l. Are devoted to software development activities.______

z. Comment:____

SECTION 2 - PROPOSITIONS

INSTRUCTIONS

Four answers are called for at the end of each of the following propositions. Check one word or phrase to complete each of the first three responses and provide a brief narrative in response to Question d.

REQUIREMENT SPECIFICATIONS

5. (Problem Nr 1)--Performance [requirement] specifications are frequently incomplete, ambiguous, inconsistent, machine dependent, and invalid. (added)

ANSWERS *

a. This problem is:

Critical	[ ]	An irritant	[ ]
Important	[ ]	Of no consequence	[ ]

Problem incorrectly worded or confusing (added) [ ]

b. This is a problem in:

Management	[ ]	Both	[ ]
Technology	[ ]	Neither	[ ]

Not a problem, unanswerable, or don't know (added) [ ]

c. This problem can be solved through improvements in:

Management	[ ]	Both	[ ]
Technology	[ ]	Neither	[ ]

Not a problem, unanswerable, or don't know (added) [ ]

d. How would (did) you solve this problem?

* In the actual survey an identical set of choices followed each of the twenty propositions.

SOFTWARE DESIGN

6. (Problem Nr 2) -- There are no decision rules for the software engineering project manager to use in selecting the correct software design techniques or tools available within the state-of-the-art.

Rewritten: Decision rules for use in selecting the correct software design techniques, equipment, and aids to be used in designing software in a software engineering project are not available.

7. (Problem Nr 3) -- There is no measure or index of "goodness" of code that can be used as an element of software design, and there is no practical way to guarantee one program is better than another.

Rewritten: Measurements or indexes of "goodness" of code that can be used as an element of software design are not avaialbe; i.e., there is no practical way to show that one program is better than another.

TESTING AND RELIABILITY

8. (Problem Nr 4) -- There are no decision rules for selecting the procedures, strategies, and tools to be used in testing software.

Rewritten: Decision rules for use in selecting the correct procedures, strategies, and tools to be used in testing software developed in a software engineering project are not available.

9. (Problem Nr 5) -- There is no measurement, or index, of reliability that can become an element of design and there is no way to predict software failure; i.e., there is no practical way to guarantee (prove) the delivered software meets a given reliability criteria.

Rewritten: Measurements or indexes of reliability that can be used as an element of software design are not available and there is no way to predict software failure; i.e., there is no practical way to show the delivered software meets a given reliability criteria.

10. (Problem Nr 6) -- There is no way to guarantee that the delivered software meets the user's requirements.

Rewritten: Methods to guarantee or warrantee that the delivered software will "work" for the user are not available.

MAINTENANCE AND MAINTAINABILITY

11. (Problem Nr 7) -- There is no measurement or index of maintainability that can become an element of software design; i.e., there is no practical way to guarantee that a given program is more maintainable than another.

Rewritten: Measurements or indexes of maintainability that can be used as an element of software design are not available; i.e., there is no practical way to show that a given program is more maintainable than another.

<u>12. (Program Nr 8)</u> -- No technical discipline exists for the design of maintainable programs.

<u>Rewritten</u>: Procedures, techniques, and strategies for designing maintainable software are not available.

<u>PLANNING</u>

<u>13. (Problem Nr 9)</u> -- It is reported that planning for software engineering projects is generally poor.

<u>Not Rewritten</u>

<u>14. (Problem Nr 10)</u> -- There is an inability to accurately estimate delivery time of a computer program.

<u>Rewritten</u>: The ability to estimate accurately the delivery time on a software development is poor.

<u>15. (Problem Nr 11)</u> -- The ability to plan for resources, particularly the number of programmers required, is poor.

<u>Rewritten</u>: The ability to estimate accurately the resources required to accomplish a software development is poor.

<u>16. (Problem Nr 12)</u> -- There is no real quality method of designing a project control plan that will enable project managers to control their project.

<u>Rewritten</u>: Procedures, methods and techniques for designing a project control system that will enable project managers to successfully control their project are not readily available.

<u>DIRECTING</u>

<u>17. (Problem Nr 13)</u> -- There are no decision rules for the selection of management techniques for software engineering project management.

<u>Rewritten</u>: Decision rules for use in selecting the correct management techniques for software engineering project management are not available.

<u>STAFFING</u>

<u>18. (Problem Nr 14)</u> -- Techniques for the selection of project managers are poor, generally resulting in poorly managed projects.

<u>Not Rewritten</u>: Procedures and techniques for the selection of project managers are poor.

<u>19. (Problem Nr 15)</u> -- There is no means of measuring with any degree of accuracy the quality of code produced by a programmer.

Rewritten: Standards and techniques for measuring the quality of performance and the quantity of production expected from programmers and data processing analysts are not available.

ORGANIZING

20. (Problem Nr 16) -- There is a poor accountability structure in most development projects, leaving some question as to who is responsible for various project functions.

Rewritten: The accountability structure in many software engineering projects is poor, leaving some question as to who is responsible for various project functions.

21. (Problem Nr 17) -- There is much consternation in industry concerning how best to organize for the accomplishment of a project (e.g., should the project be organized around the function, the project, or under a new matrix system?)

Rewritten: Decision rules for selecting the proper organizational structure, e.g., project, matrix, function, are not available.

CONTROLLING

22. (Problem Nr 18) -- It is difficult to impossible for a project manager to have the requisite visibility to be able to determine whether the project is on schedule and within cost.

Rewritten: Procedures, techniques, strategies, and aids that will provide visibility of progress (not just resources used) to the project manager are not available.

23. (Problem Nr 19) -- There is a general lack of traceability from the requirements specification to the final code.

Rewritten: Techniques and aids that will provide an acceptable means of tracing a software development from requirements to completed code are not generally available.

24. (Problem Nr 20) -- There is, in general, an inability to measure the quality of a program.

Rewritten: Success criteria for a software development is frequently inappropriate which results in poorer "quality" delivered software; i.e., not maintainable, unreliable, difficult to use, relatively undocumented, etc.

Appendix C

COMMENTS ON AND ABBREVIATIONS USED IN TABULATING THE ANSWERS

INTRODUCTION

This Appendix presents the post-survey analysis on specific questions and their responses, and lists the abbreviations/codes used on the tabulation sheet in Section 2.

To conserve space and provide a means of using a computer for analysis, all narrative answers were abbreviated and/or coded (abbreviations and codes will be called codes for the balance of this report). Because of space limitations and ease of processing, all alphanumeric codes were restricted to exactly three characters. In addition, the use of codes has an additional advantage; it effectively disguises the responses so that the participants continue to remain anonymous.

Two types of codes are used. The first type is general and applies to all questions. The second type will be applicable to specific groups of questions or subparts of questions.

PROBLEM QUESTIONS

Certain questions created more problems for the participant than others. These questions apparently confused the participants or there was disagreement with the proposition. The questions that created the strongest problems for the surveyees were Questions Number 24, 7 and to some limited degree 12, 19, and 21.

GENERAL CODES

Listed below are some general codes used with short answers or answers common to many types of questions.

CFU -- Confusing, vague, don't understand.

CPX -- Too complex to discuss in this short space.

DIS -- Disagree with question as written, not true statement.

D/K -- Don't know.

MIS -- Question not answered (supplied by author).

N/A -- Not applicable (on this project), didn't use.

N/C -- No comment (supplied by author when Part d was not answered).

NON -- No, none, or false.

NOS -- No solution (yet), hard to solve.

NOT -- Did not solve, no solution here.

NPR -- Not a problem (here), fully solvable with current techniques.

N/S -- Not specified/not selected.

OTH -- Other.

PER -- Use persistence, determination.

PRB -- This was a problem for us, we solved poorly.

RES -- Research needed.

SUB -- Subjective question, not definable.

TRU -- True statement.

UNK -- Unknown (also included "?" as a answer).

VAR -- Variable.

YES -- Yes or true.

Upon occasion the author felt it necessary to either answer the question for the participant or change his answer. In the interest of honest reporting the following codes indicate whether or not the answer was changed/contrived and the reason given. These changes codes were C01, C02, C03, and C04. C01 has the highest probability that the changed answers reflect the true answer, C02 next highest probability, C03 next and C04 the lowest probability. The change codes follow:

C01 -- This answer was supplied by the author and was based on an answer to another question or questions or other outside information (e.g., if the surveyee answered a question by saying it was the same answer as for a previous question, the answers were filled in but they were marked as code C01. If the author entered an "x" answer because the surveyee left it blank this was marked C01.

C02 -- This answer was supplied by the author to make an obvious correction to the supplied answer (e.g., in parts c and d when the surveyee checked both management and technology the answer was changed to read "both", deleting the answers to management and technology and change code C02 was used).

C03 — Answers were supplied by the author from narrative given from part d or some other source. For example, the surveyee answered "neither" as an answer to parts b and c. This was done to reduce the number of unanswered questions when it was obvious that the surveyee really did answer the question.

C04 — This answer was redirected by the author from one supplied by the surveyee to another answer. It was done only with the utmost caution and was only done if the checked answer appeared to be wrong as based on the narrative given in part d, or in a few cases, some other source such as notes in the margin. This is primarily applicable to answers checked in parts a, b, and c but disagreed with part d and it appeared the part d was correct.

SPECIFIC CODES

For Questions 1 through 4

The code "YES" on the listing opposite a question (parts a through r) indicated that the answer is "yes" or "true" as it applies to that question. If a given question has a "blank" for an answer this indicates that the surveyee answered "no", or that the answer is "false" as it pertains to that question.

For Questions 4j, k, l

The number of people employed was reported in units according to the following method. The number d(l), d(2), d(3), . . ., d(n) can be represented by d(l), d(2) X10**R, where R=N-2 and was coded for answers 4j, k, and l as d(l)d(2)R. (e.g. 10 is coded as 100, 900 is coded as 901, 6,500,000 is coded as 655, etc.).

For Questions 5 through 24

Sometimes a pseudo response, part x, was created to indicate that the participant did not provide an answer to a given question because he: 1) did not understand the question, 2) felt it did not apply to his project or organization, or 3) just did not feel like answering it. This was done so that the reader would not read a "no" when the correct answer is unknown to the author.

For Questions 5 through 24, Part a

001 -- This problem is critical.

002 -- This problem is important.

003 -- This problem is not important.

004 -- This problem is not a problem at all.

005 -- This problem is incorrectly worded or confusing (or sometimes supplied by author when surveyee did not answer).

For Questions 5 through 24, Part b

001 -- This is a problem in management.

002 -- This is a problem in technology.

003 -- This is a problem in both management and technology.

004 -- This is a problem in neither management or technology.

005 -- This is not a problem, unanswerable, or don't know (or sometimes supplied by author when surveyee did not answer).

For Questions 5 through 24, Part c

001 -- This problem can be solved through improvements in management.

002 -- This problem can be solved through improvements in technology.

003 -- This problem can be solved through improvements in both management and technology.

004 -- This problem cannot be solved through improvements in management or technology.

005 -- This is not a problem, unanswerable, or don't know (or sometimes supplied by author when surveyee did not answer).

For Questions 5 through 24, part d

In answering part "d" of Questions 5 through 24 a different approach must be used. These were narrative questions calling for opinions and/or lessons learned by the surveyee. The answers are frequently verbose, very opinionated, and represent some lesson learned the hard way. The answers were grouped under the management functions of planning, organizing, staffing, directing and controlling and the technical functions of requirements, design, develop and code, and index of performance. The first letter of the three character code indicates the grouping.

C -- Controlling

D -- Directing

I -- Index of performance

O -- Organizing

P -- Planning

R -- Requirements

S -- Staffing

X -- Design, develop and code

CONTROLLING CODES

CMa. Track manhours expended vs budget (cost/performance charts) (rate charting) and compare with documented evidence of progress.

b. Establish that if a software system meets requirements (or is at least useable) it has quality.

c. Monitor schedule closely.

d. Provide visible measurable milestones, coupled with reporting, reviews or walk throughs.

e. Apply judgement to determine if software meets quality.

f. Use an automatic monitoring system and/or other tools.

g. Establish a quality assurance function.

h. Assign more personnel to project control.

i. Document and review task description, agree to schedule and output, and make personnel responsible for work assigned.

j. Put numbers on subjective measurements.

k. Control changes (configuration management plan).

l. Use walk throughs.

m. Use (frequent) reviews and reports.

n. Use milestones.

o. Keep the customer informed.

p. Document all delivered products.

q. Have a (separate) project control function.

r. Measure against the development plan.

y. "Quality not specific enough, must be defined.

CTa. Develop guide books.

b. Establish good tracing techniques (matrix) from requirements to design with appropriate test criteria.

c. Use a tracking tool such as threads or a similar technique.

d. Establish and adhere to programming standards, software design methodology.

e. Use a technical control board.

f. Use a work breakdown structure (WBS).

g. Do extensive and thourough testing and verifying.

h. Check test results and accuracy of interfaces against known results.

i. Find a method of testing and measuring "quality."

DIRECTING CODES

DMa. Institute an earned value type scheme with meaningful work measurement parameters.

b. Select a method applicable to project, plan for it and stick with it.

c. Treat each case differently and select the best method available.

d. Develop a list (survey) of techniques and select from this list to meet the requirement.

e. Use the same techniques that worked in any other (i.e., hardware) development.

f. Obtain through trial and error.

z. Any technique is valid.

INDEX CODES

IMa. Review constantly (involve the user with) the software development process.

b. Use the many (best) indexes available.

c. Establish an (independent) software testing and validation effort.

d. Keep maintainable statistics.

y. Not necessary if software meets design requirement as demonstrated through tests (or if it is not specified).

ITa. Design parallel software, test and then select best one.

b. Build design verification tools.

c. Use proof of correctness techniques.

d. Define and develop software development metrics.

e. Develop tools/techniques to measure metrics.

f. Determine the software's capability for easy change.

g. Use simulation and prototyping techniques.

h. Establish a MTBF for software.

i. Use fail-soft hardware/software systems.

j. Do a thorough job of reviewing, testing and verifying.

k. Test, if it works, it is a good code.

ORGANIZING CODES

OMa. Establish (improve) the proper authority of the organization (define authority and tasks) (early in the project) (document the decision).

b. Require the acceptance of responsibility (by project manager).

c. Use a functional organization.

d. Use a project organization.

e. Use a matrix organization.

f. Give responsibility to programmers for implementation, interrogation and testing (obtain their concurrence and understanding).

g. Use programming team concept.

h. Train personnel in how to develop and follow a proper organizational structure.

i. Place the project organization under one manager.

z. Use any (and all) methods that work.

PLANNING CODES

PMa. Develop (early) correct (detailed) planning for the project that accomplishes the task (schedule, budget, etc.).

b. Establish a (more) formal project phase (function).

c. Understand the requirement specification (first).

d. Plan for better aids (PERT, CPM, etc.).

e. Initiate R&D in project management.

f. Understand the (technical) problem by management.

g. Involve everybody connected (customer, developer, technicians, etc.) with the development.

h. Agree on task to be performed (by everybody), improve communication, and select common goals.

i. Control the factors outside of the project managers control.

j. Break planning down into lower levels of activities.

k. Schedule the project better.

l. Have software development project follow proven engineering standards (like hardware).

m. Execute, use and enforce the plan.

n. Documentate the plan better.

o. Break system down into detailed levels.

PLANNING CODES (continued)

p. Review (by senior management) and update throughout the project.

q. Have a fall back position.

r. Use planning tools, techniques and procedures (used by similar successful projects) (collect historical data).

w. And will increase the development costs.

PSa. Allow enough time to do a good job of estimating.

b. Use a conservative estimate with contingencies.

c. Use best methods (rules) (available).

d. Size the work effort and complexity accurately and realistically.

e. Use bottom up planning.

f. Develop and use a detailed milestone schedule.

g. Have greater involvement by programmer/analyst.

h. Research methods of estimating schedule and cost.

i. Research and review past projects (build data base on cost/schedule data).

j. Review the schedule constantly.

k. Staff and fund realistically.

l. Use (documented) work performance measurement (structure) for manloading.

y. Not a problem, a low estimate was made.

REQUIREMENT CODES

RMa. Have a (more) formal requirements phase with adequate time for preparation.

b. Deliver on time (before starting design).

c. Review at the beginning and throughout the project (to determine completeness) (for "Quality").

d. Control and/or base line the specifications.

e. Agree on specification between all organizations involved (customer, developer, management, technician, etc.).

f. Develop specification independent of project.

g. Develop jointly (or involve) the customer, developer, management, technicians, etc.; work with/get customers input.

y. "Quality" was not a requirement.

REQUIREMENT CODES (continued)

RTa. Establish specification that will accomplish its intended function of specifying the customer's requirements (before beginning design).

b. Verify, analyse, measure, and test the software (independently).

c. Use formal (modern) requirement generation techniques and/or methodology (specification languages, top-down design, standards, etc.).

d. Accumulate more specific details and complete specifications.

e. Have clear, readable, and understandable specifications.

f. Have unambiguous and constant specification.

g. Have useable, realistic, and valid specifications.

h. Redesign system (if necessary).

i. Analyze the problem (simulation techniques) (prototyping) thouroughly and carefully.

STAFFING CODES

SAa. Review and monitor programmer product and documentation by knowledgeable (peer) people.

b. Train and educate the customer.

c. Keep records on each programmer.

d. Staff with competent programmers adequately.

e. Establish, use and enforce best personnel standards available.

f. Use best qualified programmers available.

g. Increase staff as needed.

h. Match jobs to personnel.

i. Recruit and hire qualified programmers appropriate to the job.

j. Use knowledgeable, experienced and trained project managers to lead, manage and communicate.

k. Train and educate programmers (to produce good code).

SMa. Select managers based on experience, managerial and technical abilities (not availability).

b. Define the attributes of a good manager (and apply).

c. Design test to test potential project managers.

d. Review past project to establish what project manager quality will improve chance for success.

STAFFING CODES (continued)

e. Let the project manager manage.

f. Do not use "old school", bottom-up managers.

g. Have the reviewing authority also be a good manager.

h. Select managers who have been successful in similar type projects (experienced) (even if project was smaller).

i. Match the project requirements to project manager's background.

j. Do not push/use technical personnel into management when they are not qualified.

k. Provide appropriate training for project managers.

l. Provide OJT (apprenticeship) for project analysts to train them to be managers.

DEVELOPMENT, CODING CODES

XMa. Making it a (more) formal phase.

b. Use competent experienced technicians to select appropriate tools, techniques, procedures, etc.

c. Keep a library of tools, procedures and techniques available for selection.

d. Use those rules, policies and procedures you do know (or through consultation with others) that will do job and select the appropriate ones.

e. Develop a list (survey) of software development tools, techniques, procedures and select from this list to match the current needs.

f. Enforce procedures, techniques, tools selected.

g. Use procedures, tools and testing procedures used by a similar successful system.

y. All programs are maintainable (some easier than other).

z. Any method will do in software design, tools and technology if the software meets design requirements.

XTa. Research in developing decision criteria (metrics) to enable managers to evaluate and select software tools.

b. Develop or establish your own software design tools, techniques, procedures, etc., (to the best of your ability) for designing and testing software.

c. Research and develop software development standards, procedures, test procedures.

DEVELOPMENT, CODING CODES (continued)

d. Analyze the software development requirement and select the proper tools, techniques, procedures, etc.

e. Allow adequate design flexibility.

f. Use top-down structured programming, modern programming and design approach.

g. Use an integrated approach to software development and testing.

h. Bench mark or simulate a subset of the software design to select proper tools.

i. Select a specific procedure.

j. Use bottom-up testing approach.

k. Use the same procedure as hardware.

l. Use good software design standards that if met will develop "good" reliable, maintainable, usable code.

m. Use good documentation standards.

n. Use small programs.

Appendix D

NARRATIVE RESPONSES TO PART "D" QUESTIONS

INTRODUCTION

This section deals with the actual mostly unaltered answers provided to the Part "d" portion of each question. Part "d" asks "How would (did) you solve this problem?" This, of course, required a narrative type respor :e. Unfortunately narrative type response are not amenable to reduction and analysis on a computer. Therefore, all of these narrative responses were defined and grouped by type. Each grouping was given a code and this code is reported on the tabulation sheet in Section 2 (the code itself can be found in Appendix C). However, since this reduction of comment to code destroyed some of the richness of prose, the author felt it was worth-while to include this "verbal" section in the report.

The answers as they appear in the following pages have been "cleaned up" to assure anonymity from the standpoint of author, firm, and project. Identical or nearly identical responses have been eliminated as have incomplete (incompleteable) sentences and "one-worders." With the exception of the "clean up" and correction of the most obvious spelling and punctuation errors, those responses included in the following pages are as received and though they do not in every instance answer the question asked, they do relate to the subject. As an aside, we make no claim t· total understanding of every response.

The narrative answers to the "Part d" questions are all grouped under specific Problems 1 through 20 (Questions 5 through 24). Also included with the Part "d" answers is how the surveyee answered Parts a, b and c. As you realize from reading Appendix C in this report, Part a is coded 1 through 5, Part b is coded 1 through 5, and Part c is coded 1 through 5. If, for example, the surveyee answered 1 for Part a, 2 for Part b, and 4 for Part c then the code 124 precedes the narrative answer in parenthesis in front of the answer. This is to give the reader a feeling for the participants attitude as he wrote the answer.

Also whether or not the respondent marked "would" or "did" is placed in parenthesis and preceeds the response to Part "d". Very few bothered to indicate. Also the author feels obligated to call to your attention that some of the codes preceding the part "d" answers were not necessarily the ones the participant put down. As previously described in Appendix C some of the answers were changed or contrived. It is the changed or contrived answers to parts a, b and c that appear here. We do not believe that any significant amount of information is lost in this method of reporting.

QUESTION 5. Problem - Performance specifications are frequently incomplete, ambiguous, inconsistent, machine dependent, and invalid.

SOLUTIONS

(111) Encourage/force specification development jointly by customer and analyst.

(111) Development and enforcement of formal standards for specification generation with frequent reviews by management and involved technical personnel.

(111) Require complete specifications before signing/negotiating contract; or use phased contract approach -- phase (1) prepare specifications; (2) implement.

(111) (1) Assign specialists to insure performance specifications are complete, consistent, non-ambiguous, and valid and that the specifications can be met within the budget and schedule commitments made under contractual requirements. (2) Provide training in specifications writing; attempt to drive hardware-like specifications (one requirement, one paragraph); testable; eliminate equations, descriptions, and BS. (3) Rework MIL-STD-483 to the above. (4) Avoid HIPO-type and "formal" specifications. This is a waste; better techniques available for real time.

(111) Recognize performance specifications inadequacy as directly effecting delivery and react as if customer delivery was late.

(111) Work with the customer to obtain an agreement on the performance requirements.

(111) Depends upon when problems are uncovered. If early, complete SRR should provide full visibility followed by complete requirements allocation by SDR.

(111) More calendar time for specification writing. Schedule spe ification delivery to programmers with same detail that output sot ire is scheduled. Give the people writing specifications time to understand the entire problem before being forced to get something on paper.

(113) (did) (1) Primarily a management problem due to lack of insistence on qualitative and quantitative specifications, which result in no measurement criteria. (2) An obtainable performance objective must be agreed upon early in the program with minimal modifications allowed during the development.

(122) Use formal requirement methodology.

(132) Better performance specifications.

(133) (would) Make efficient use of best technical personnel who could be found.

(133) Go to the root of the problem by evaluating the source of the specifications.

(133) Top-down design with mandated traceability among levels of specifications and code.

(133) By performing requirement analysis and rewriting the requirements as a result; also working closely with customer.

(133) Involvement of both the designer and the developer during performance specifications.

(133) Establish the approach of using a software integration engineer who understands problem physics and a computer program implementer of physics algorithm.

(133) Team with customers/system engineer teams to build specification ICD's and use modern specification definition techniques.

(133) We were required to use MIL-STD-483 and write B5/C5 specifications. Part of the difficulty is that the outline for B5 specifications <u>erroneously</u> puts forth the proposition that equations are requirements. I personally fought this battle, and won; we do not today have a single equation in our requirement specifications. The problem, of course, is that an equation gives the impression to a reader of detail and precision. Unfortunately, an equation is just a <u>description</u>, couched in algebraic language rather than English, and has nothing to do with <u>requirements</u>.

(133) 1. Get software managers with technical (application) capabilities; 2. Get an executable or analyzable statement of the requirements.

(133) Have software development team participate in requirements definition, start software development later in system cycle, use parallel requirements definitions/software development periods with formal change management starting from "base line" definition.

(133) 1. Start software systems engineering with concept formulation (hardware and software). 2. Improve the method of sizing software tasks.

(133) Redesign system.

(133) A great deal of communication with the customer prior to project go-ahead.

(133) More time to prepare formal and complete specifications.

(133) Software participation in specification writing and use of simulation modeling in validating performance specifications.

(211) Design reviews to identify base line.

(211) Manager must work with customer to develop better specifications.

(211) Assign specification preparation to an independent, well staffed organization (rather than programmer manager's staff).

(211) The source of the problem is inadequate technical understanding of the problem by the customer, usually the government. The solution is to provide an environment that will attract and keep technically competent personnel.

(211) The problem can be prevented from the very beginning by taking precautionary procedures in management, such as an early decision, assigning responsible persons to handle, careful checkups and testing, etc.

(2XX) Establish an independent verification of the performance specifications to assure single interpretation and verification of the requirements.

(231) Provide an effective communication path between management and technical personnel regarding specifications.

(233) Better analysis of problem in deriving requirement specifications.

(233) Work closely with customer to resolve discrepancies.

(233) Suffered because of this problem.

(233) Attempted to base performance requirements on existing, documented figures.

(233) (would) Start project specifications activities sooner, to give people more time.

(233) Establish requirements traceability in an explicit and positive manner, e.g., traceability matrix.

(233) More time should be spent on this, programming support and actual psuedo-program testing on computer.

(233) System engineering team (customer, developer, management, members) developed specifications.

(233) Require more detailed performance specifications early in program.

(233) Use specification languages and specification guides.

(244) Although customer did not generate specifications, development team created realistic goals.

(311) Require clear and complete statement of requirements and specifications. Iteration between those establishing specifications and those who have to meet them.

(311) Change Air Force policy in initiation and manning of projects.

(344) By building a close, friendly relationship with the user to understand his real needs and problems.

QUESTION 6. Problem - There are no decision rules for the software engineering project manager to use in selecting the software design techniques or tools available within the state-of-the-art.

SOLUTIONS

(111) (did) Implement policy and procedures that define a minimum set of techniques and tools for all projects.

(111) Technically up-to-date technical management that has the background to know/understand the tools, equipment, and the timing. It takes a commitment to planning to get it right.

(133) All these new methods have advocates and they often make contradictory claims. A survey must be made and one selected.

(133) Either team-up with a professional or shape up ourselves.

(133) Establish team to select the best tools/methods. Try them and use them.

(133) This problem occurs when the engineering project manager does not know software design techniques. So it is important to assign someone who knows software techniques well enough and can make the decision.

(133) Take the approach for optimum performance and if it is of a high technical risk, always have a back-up position prepared.

(211) Continued education at company expense.

(211) The absence of rules is not necessarily a problem if the project manager has previous experience. This problem equates to having experienced project managers, which is a training issue, or one of personnel development. If no techniques or tools existed, it would be a technology issue.

(211) More education of management into modern programming tools and techniques.

(211) Select one the staff knows best.

(213) Training for upper level ADP management.

(222) Establish known rules.

(222) Software was designed according to available expertise.

(222) Developed software.

(222) Solving problems is difficult; practice makes it easier; give tools which help state the problem.

(222) It is extremely difficult to "cook book" this area -- requirements differ for each new job.

(233) Establish decision criteria to enable evaluation and selection of software tools. Develop or adapt standards and enforce utilization of software design techniques deemed optimum.

(233) Compare the requirements of system against approaches taken on similar projects and their successes.

(233) Keep a library of tools; rely on technical performing organization to recommend tools; interchange experience between projects.

(233) Used available tools and designed unique ones within budget constraints.

(233) Survey all possible software design tools/techniques available for application to the particular project. If none are available pursue establishing requirements through management functions and develop necessary tools.

(233) Spend funds to develop proper tools in hopes of pay-back in software development cost.

(233) Arbitrarily select a constant set of techniques to use (structured design, top-down implementation, and HIPO's for modules) and push the technique.

(233) Survey of current practice.

(311) Meet with design leaders.

(311) Select tools on basis of availability, capability and the experience of others. If several programs will need tools, assign a person to develop decision rules and index to existing tools.

(314) This is a situation which occurs in all state-of-the-art technology. One must keep current and learn to live with it.

(322) Applied structured, top-down design approaches.

(322) Use techniques proven on simulation type applications.

(322) (did) Rely on competence of individuals.

(333) Research and, more important, education of personnel involved in software design.

(333) (did) This project was completed before the current emphasis on techniques gained general acceptance.

(333) Analyze the system requirements to determine the design approach and software tools needed to satisfy these requirements.

(344) Had adequate design flexibility.

(411) (would) Simply choose one; they all work. The real crux of the matter is personnel quality more than design tools.

(455) This may not be a problem at all and may simply reflect the confidence of the customer that management will select the best tools available at the time required.

(455) The techniques and tools vary a great deal from project to project and no set of rules is good for all cases.

(555) I don't understand the question. It is not clear to me that software design techniques are an important issue if the project manager understands what he is supposed to produce.

QUESTION 7. Problem - There is no measure or index of "goodness" of code that can be used as an element of design, and there is no practical way to guarantee one program is better than another.

SOLUTIONS

(122) Develop automated tools for design verification.

(122) We are analyzing HOS as possible answer.

(211) Much more testing at several levels (unit, local integration, global integration, system integration, project integration). The best way to measure code "goodness" is to test it thoroughly.

(222) Perform tests with various alternative programs, analyze results, and select the most positive one.

(222) Rigorous testing will determine that the program meets design requirements and not affect system performance.

(222) (did) Establish some after a literture search and discussions with experienced people/projects.

(222) Define criteria for measuring relative goodness, develop qualitative measures, and apply them. The criteria and measurements will differ with different types of programs; they will not be universal. Doing this will be expensive, but worthwhile.

(222) Very hit and miss.

(222) Education. Discussion. Even today we can write "better" FORTRAN programs. Of course, part of "better" comes about because of changed definitions of what is better. In early days we fought for execution efficiency and memory usage minimizations. Today we look for better programs in the sense of understanding, readibility, etc. This problem will be self-solving as constraints of technology are removed (e.g., faster machines, cheaper memories).

(223) (did) Implement policies and procedures that define and require the use of metrics for software development.

(223) Simple (or short) elegant coding can be produced by intelligent programmer who has logical mind to do so. Even then, there is no way to guarantee one program is better than the other. Hence, select carefully the right programmer (who can make the job sell done) to do the work.

(232) Use more software simulation techniques.

(233) "Goodness" must be defined. If standards are applied (assuming they are "good") then program designs which meet these standards, meet system requirements, and work "correctly" can be evaluated against these criteria.

(233) (did) A thorough and continuous review process.

(233) Define criteria beforehand, and build tools to measure it at keypoint in the software development.

(233) The problem may be relieved through the adherence to programming standards within the project.

(233) The importance is job dependent "goodness" can be "time" dependent. "Memory size" dependent, etc. Must be solved by better communications of the situation at hand.

(322) Was largely unsolved so far as code efficiency was concerned.

(322) Code review; using structured programming conventions.

(322) Rely on competence of individuals.

(322) Get smart programmers.

(322) There are methods of measuring "goodness" of code but they require duplication of effort, at same level, and are thus costly.

(325) Don't worry about measurement of goodness; test to see that it does as required.

(333) It can only be solved through academic and on the job training courses, together with discipline and careful review. It will drive the development cost upward.

(422) Within limits, program variation between programmers is of no consequence if the product will do the job. The programmer should be exposed to suggested "good" techniques.

(432) Having a goodness factor is somewhat academic. Use rigorous testing under system performance variations. Qualification of goodness will not be directly relatable to any single point, software failure.

(444) All code generated is measured by its performance in accomplishing specification requirements which is the only guarantee provided. The measure or guarantee that one program is better than another can only be derived from specific cases (i.e., one program produces more optimum code, or has fewer bugs than another). Top-down design and structured programming can relieve the number of bugs, but increase the amount of code.

(444) Goodness depends upon specific objectives. These shift from project to project. There are many "indices" of goodness, e.g., size, speed, maintainability, documentation.

(444) Programs are equally "acceptable" if they meet requirements.

(455) There are indeces of goodness; does it meet specification; is it maintainable; well commented; understandable.

(455) This has always been considered black or white -- which translates to good or bad. If the system function was as desired, it was good. Otherwise it was bad and there was nothing in between.

(555) There is no substitute for qualified people in whom the project manager has confidence. Problem statement is confusing. Why must one program be better than another if both meet specifications?

(555) This problem has many facets and may begin during analysis at the time where processing algorithms are selected, or they may be strictly a reflection of inexperience associated with selection of coding techniques. How do you define "goodness"?

QUESTION 8. Problem - There are no decision rules for selecting the procedures, strategies, and tools to be used in testing software.

SOLUTIONS

(111) Spend much more time developing rules and procedures for testing and in testing itself.

(111) Used the approach that each had the most confidence and success with/over their previous experiences.

(122) Have not solved it yet. I think the design problem is more critical.

(133) Either team with a program -- shape up ourselves.

(133) A bench mark set of problems was selected as a stability test and used to help detect problems.

(133) Perform trade studies to determine most cost effective approach to be used for testing.

(133) Rigorous operational simulation with software and hardware instrumentation.

(133) Formalize validation procedures; defined as part of contract.

(133) Poorly.

(133) Get smart systems testers that are not under software manager's control.

(211) (did) Implementation of policies and procedures.

(211) Testing is unique to the type of software generated (i.e., control, support and test). Develop accurate procedures (including necessary tools) for testing software prior to release (e.g., develop a complete test case for a compiler and insure it is ran without error prior to any release).

(211) Define test criteria for each program based on performance specifications.

(211) The absence of rules is not necessarily a problem, if the project manager has previous experience. This problem equates to having experienced project managers, which is a training issue or one of personnel development. If no techniques or tools existed, it would be a technology issue.

(211) (did) Select levels of testing and test criteria that match costs/schedules/resources and tools availability constraints of the project.

(211) Send the manager to get training courses.

(222) Establish basic rules.

(222) Test by sampling input space, using formally reviewed acceptance test.

(222) Spend lots of time testing.

(222) (did) Separate test team. Exhaustive test at every level.

(233) Made assumptions, conducted tests based on experience and recommendations of key senior people.

(233) Establish criteria upon which such test tools, procedures, etc., can be evaluated in terms of project needs. This involves education and the latest development in software disciplines.

(233) I established them on a specific project.

(233) A funded program does not usually allow development of general purpose tools. Since flight software is usually programmed in assembly language, the procedures, strategies and tools are unique to the computer and problem.

(233) Write testable criteria in requirements. Define development approach to use available tools, make sure tools are ready at test time, apply change management to tools also.

(233) Define criteria for measuring, testing, developing qualitative measures and apply them. The criteria and measure will differ with different types of programs; they will not be universal. Doing this will be expensive, but worthwhile.

(233) Test procedures were established at the initiation of the project to insure that all requirements were met. These procedures were then enforced by management.

(233) Specific R&D activity.

(255) I either don't understand the question or the problem. I don't see mystery in software and find the range of techniques of management to be fully useful.

(311) Devote more time to planning test phase. Assign an independent test team.

(324) Decision rules depend upon specific objectives. These shift from project to project. There are many decision rules. We do not work in a static field.

(333) Experience is still a more satisfactory attribute than any programmer rules.

(422) We test systems for proper performance and run demonstrations. The computer program integrity is verified at this time.

(444) Develop on individual case basis.

(455) There are testing aids and disciplines.

(455) Procedures, strategies, and tools reflect the nature of testing requirements relative to performance specifications. Where did you get these magic decision rules?

QUESTION 9. Problem - There is no measurement, or index, of reliability that can become an element of design and there is no way to predict software failure; i.e., there is no practical way to guarantee (prove) the delivered software meets a given design criteria.

SOLUTIONS

(113) (did) Provided the most stringent tests possible within the allotted time.

(121) The manager (customer type) has not insisted that his people impose requirements in an manner which permits the product to be tested properly!

(122) Establish minimum design objectives to be met.

(122) Improved design techniques; automated design verification.

(122) (did) Extensive testing at every level.

(132) Design review board.

(133) Exhaustive testing!

(133) Through a combination of programming standards and enforced testing procedure.

(133) Better design techniques, longer testing in "real" environment.

(133) Spend adequate time testing.

(211) Insure specifications are complete, accurate, and understandable. Design the software around the specifications using top-down structured design, high order language and perform testing in accordance with approved procedures.

(211) Again, the way to verify reliability is in testing.

(214) Large programs are impractical to guarantee against failure by any method I know of.

(222) Continue data gathering and data analysis.

(222) I see it coming as the next big push in software technology. We are spending IR&D monies in this area to collect data on flight test -- released program flight hours vs errors. Also looking at predictive models.

(222) Rigorous testing and simulation to guarantee software performance.

(222) Proof of correctness.

(222) Not important on this project.

(222) (did) Perform exhaustive system testing.

(223) I established them on a specific project.

(232) Solved usually by fault insertion of "test data."

(233) There are ways of doing these things, but they take additional time from actual implementation which people are not usually willing to pay for.

(233) (did) Adoption of a proven reliability measurement scheme.

(233) Used a criteria that the software "seems to work" in a production environment with non-computer staff; it's reliability is therefore adequate.

(233) (would) When the customer puts a reliability (or any other defined index) in the system requirements, technology will be developed to satisfy the requirements.

(233) Much reliability is inherent in the completeness of the analysis and design effort from performance specifications to coding, as well as detailed test plans which should be part of the design phase.

(233) (would) Impose top-down, structured (design simplification) techniques. Require fail-soft overload (out of specification) processing and recovery from intermittent hardware faults. Conduct detailed reviews to monitor adherence to established criteria.

(233) Reliability is "designed in" and achieved through rigid design procedures. Also, having a reliability measure is somewhat academic. Rigorous testing under system performance variations. Qualification of reliability will not be directly relatable to any single point in software failure.

(233) Get good people to work on the job. The emphasis on all software development must be on the quality of the people rather than on the procedures for doing a job with a large number of people of limited competence, and inventing elaborate procedures for managing, monitoring, and correcting the feeble and expensive efforts.

(322) We test to SRS.

(333) Extensive comparison of design walk-through results with software output/results.

(333) Test the software thoroughly before committing to its use as the residual unknowns are minimal and acceptable. Build in failure recovery methods for critical problems.

(344) An initial decision on how much time and effort is to be spent on such assurances. It will vary from case to case.

(444) Testing: test by sampling input space using formally reviewed acceptance test.

(455) In the application which I have worked for 10 years this "reliability" thing. . . .

(555) I don't think the question is correct. I think it is possible.

QUESTION 10. Problem - There is no way to guarantee that the delivered software meets the user's requirements.

SOLUTIONS

(111) Design and test plans must be continually reviewed and evaluated in terms of user requirements. Enforcement of such policy will guarantee meeting requirements to within a very small tolerance.

(111) Precise, detailed statements of requirements. Through acceptance testing.

(111) User requirements must be specified and test program must show that all requirements in specifications are satisfied.

(111) Establish a series of design reviews (both hardware and software) with the user to insure proper communications are established. Keep communciation channels open and allow user walk-throughs when appropriate. Try to set up early demonstrations or prototype useage for user evaluation during development.

(111) Traceability from approved requirements through all levels of documentation.

(111) User should get involved with the development of software. If possible, he should participate in the software review periodically, to make sure that the software will meet his requirements.

(111) Earlier and more participation of users in requirements. Participation of users in testing.

(112) System testing.

(113) Improve the software requirements to insure that they are all testable. Rigorous testing over complete range of data values.

(121) The manager (customer type) has not insisted that his people impose requirements in a manner that permits the product to be tested properly!

(122) (did) Intensive testing.

(122) (did) Extensive testing.

(133) Have project reviews on a regular basis with the user, and try to resolve differences between the specifications and implementation problems.

(133) All paths through software couldn't be checked (time constraints); critical paths tested to specification. Final test equals production.

(133) Establish traceability.

(133) Establish cross checks on algorithm development and validation simulations. Exercise flight code over extreme excursions with as much flight hardware as possible.

(133) Prepare an SRS and review with user.

(133) (did) Establishment of a software verification and validation functional organization.

(133) Executable/analyzable statement of requirements.

(133) User represented in design activities.

(133) By achieving a firm set of agreed-to requirements and enforcing a comprehensive test program, this problem can be relieved.

(133) Greater involvement of the user community. In most instances, to gain maximum benefit, the user must have more than a "black box" concept of what the computer system can do for him.

(133) Redesign to users requirements those that were practical.

(144) Communication.

(211) Involve user in requirements definition and review. However, user often doesn't know what he wants.

(213) Testing, both functional and integrated is the key. If requirements are clearly stated they can be tested for. The depth of testing is a function of time and cost.

(214) Large programs are impractical to guarantee against failure by any methods I know of.

(222) Major requirements were checked.

(222) All major requirements were verified.

(222) No mystery here -- it's tied into "requirement" and testing. As part of requirements definition, test definitions can be solidly tied down. It can work as well as hardware testing; get the mystery and crap out of the requirements. Implement a scheme for achieving concurrent documentation. We did this by adapting the C5 format (and by moving equations from the B4 to the C5). I feel strongly that the outline for B5/C5 specifications is contributing to the problem of requirements definition by substituting (or mistaking) detailed descriptions for requirements. For evidence, see the Fire Control Computer OFP.

(232) Test the software extensively before delivery.

(233) Write good specifications; use pilot projects; have user participate in development review and prototype evaluation; keep developers around for a "maintenance" phase.

(233) Keep software people in contact with user representation.

(244) The primary problem is in communications. The ability to state requirements on a system or user level.

(311) Communication with user.

(311) Develop test plan.

(322) Thorough testing does an adequate job of proving (or disapproving) that requirements are met.

(411) Communications and user involvement.

(444) (did) Test specifications, test planning, and test procedures reviewed by the user.

(455) Not a problem because we involved the user in developing the requirements and throughout the development process.

(555) I disagree with the premise. Get good people to work on the job. The emphasis on all software development must be on the quality of the people rather than on procedures for doing a job.

QUESTION 11. Problem - There is no measurement or index of maintainability that can become an element of design; i.e., there is no practical way to guarantee that a given program is more maintainable than another.

SOLUTIONS

(211) Well-written documentation, small modules with minimal interdependence , structured design, well commented listing, and a sound control system will achieve satisfactory maintainability.

(211) The key to maintainability is documentation. Stricter rules for generating documentation and commenting code. Enforce concurrent code/documentation.

(211) The software should be designed to be understandable by other programmers with more suitable comments, statements, and should be feasible with them.

(211) More complete and automated documentation.

(213) Insure that software standards are maintained and that software is properly documented.

(214) Use programming guidelines that include maintainability objectives.

(222) Actually test different programs and introduce possible bugs, then evaluate which program is easier to "fix".

(222) Study all new cncepts on software design, keeping abreast of actual concepts put in use to insure maintainability. Establish indexes to implement those concepts that prove to develop software that is easily maintained (e.g., HOL, modular design, etc.).

(222) Presently doing independent research into maintainability metrics.

(222) Yes -- except to standardize on selection of higher order languages, the use of constructs, the nature of documentation and the simplicity of interfaces and formats.

(222) Use structured design.

(222) Train customer adequately.

(233) Establish traceability.

(233) Wait and see. Otherwise, must maximize effort on documentation on all levels.

(233) Use good people to begin with, document well, and emphasize personnel continuity.

(233) Establish design guidelines and forced conformance.

(233) Allocate funds for warrantee and rigorous control and documentation procedures.

(233) This problem was reduced through structured design and enforced programming standards. Documentation standards must also be enforced.

(233) Structured programming methods.

(311) Hire the best programmers/analysts you can: you'll get reasonable programs.

(311) Require better documentation. Improved programming techniques.

(311) Maintainability can be measured on a project after delivery. This used as correcter on next project.

(322) Review alternate technical approaches with respect to maintainability during early development phases.

(322) Evaluate code and documentation. System capabilities do change.

(333) Establish heuristic measures of maintainability and apply them.

(333) Improve documentation techniques.

(333) Conduct trade studies in beginning to determine what maintenance oriented requirements to incorporate into requirements and pay for them; use best software technology available to ease maintenance tasks.

(344) An index can be developed by using the ratio of write to debug time. Usually the higher the ratio the better the maintainability.

(355) I'm not familiar with the corresponding hardware indices, but will say that good design is essential to "good" maintainability. And since I'm stuck on requirements and concurrent documentation, I think the answer is there.

(433) Relative index ratings are for conversation and don't really qualify anything except someone's subjective rating.

(444) This was a feasible study.

(444) (would) Simplify design (structured/top-down) complete documentation (3560.1 plus auto-generated as built documents).

(455) There are maintainability indeces, in particular, seeding of errors, followed by the accrual of maintenance statistics.

QUESTION 12. Problem - No technical discipline exists for the design of maintainable programs.

SOLUTIONS

(111) Solution (if that is the word) is longevity of staff on project.

(133) Again, structured design and programming standards will relieve this situation. Also, the design should be as simple as possible.

(211) Well written documentation, small modules with minimal interdependence, structured design, well commented listing, and a sound control system will achieve satisfactory maintainability. Basically, the principles involved are no different than those used for developing maintainable hardware.

(211) Statusing techniques against budget and/or schedule are obviously inadequate. Also, the status reports of optimistic programmers must be carefully looked at.

(211) More complete and automated documentation.

(222) Establish minimum maintainability standards.

(222) Presently doing independent research into maintainability metrics.

(222) Perform analysis to determine technical disciplines that allow for the design of maintainable programs. Use those disciplines (e.g., HOL, modular programming, top-down design and testing) that will insure proper design.

(222) Use structured design.

(222) Train customer.

(222) (would) Require top-down design and modularization by function.

(233) Define problem. Set up standards/guides.

(233) Establish traceability.

(233) Use modern programming techniques and produce concurrent documentation.

(233) Structured programming techniques.

(233) Establish design guidelines and forced conformance.

(233) Technology can provide better program documentation language, etc., however, management must insist on structure and be willing to live with its cost and schedule impact.

(311) Adherence to proper software standards.

(322) Research needed.

(322) Develop guidelines for use in local programs.

(333) Standards.

(333) Conduct trade studies in beginning to determine what maintenance oriented requirements to incorporate into requirements and pay for them; use best software technology available to ease maintenance task.

(333) Funds to establish a plan.

(333) Structured programs tend to be more maintainable.

(433) Maintainability is "designed in" through appropriate levels of modularity and documentation completeness. A separate technical discipline is not necessary.

(444) (would) Simplify design (structured/top-down). Complete documentation.

(455) All programs are "maintainable." Some are easier to maintain than others. We are aware of many of the things which enhance maintainability, the question is quality versus cost.

(455) Maintainability requirements are a function of end use. It can be designed into software if required.

(555) Not true; the structure of modules and data base can be designed following rules such as Constantine's or Yourdon's.

(555) I don't think the question is correct. I think it is possible.

(555) By defining/identifying specifics. You can't treat sickness until you identify the particulars.

QUESTION 13. Problem - It is reported that planning for software engineering projects is generally poor.

SOLUTIONS

(111) Do it right! Train managers! I established visibility techniques with daily meetings and planning papers.

(111) (did) Developed a plan, updated it monthly, and stuck to it.

(111) Improved communication between programmer/analyst and management. More comprehensive reviews; agreement on goals.

(111) More extensive and more structured planning. Greater involvement of analyst/programmer personnel. Provision for detailed review of plans.

(111) Suffered due to poor planning.

(111) Review recent and on-going projects.

(111) In future projects -- more effort should be spent in planning.

(111) (did) Establishment of a requirement for detailed development plan for all projects.

(111) Early establishment of development plans with a series of review milestones to monitor progress.

(111) Develop a software engineering plan to identify all elements, schedules, budgets, and milestones to be met. Provide all personnel with a copy and convey full understanding to all persons involved. In essence, plan the project and execute the plan.

(111) Train managers of software projects in problem solving/software technology and management.

(111) Training of managers.

(111) Plan with the same intensity as that used for hardware.

(111) Spend more time planning and plan in more detail. Monitor and enforce plan.

(113) Development of automatic aids (e.g., CPM, PERT); training of project people in various planning methods, formal classes.

(114) I think the problem isn't in the plan, but the knowing where you are. Tangible measurement of progress is the key and concurrent documentation is the answer. Reason - software in development is just documentaion. No documentation, no software, no progress, no plan.

(131) More time must be devoted to understanding the problem (requirements) prior to implementation planning. The most knowledgeable people must be used in planning and scheduling after they have been given sufficient time to study the requirements. Frequently planning is done in a few days picking numbers from a hat. This is done so as not to distract technical personnel from the present assignment, but results in poor planning.

(131) Management guidebooks should be prepared.

(133) The problem experienced was not planning but execution, it was made extremely difficult by influences outside the project manager's control.

(133) In depth schedule preparation.

(133) This problem was reduced by coordinating the initial design effort with the project planning phase to insure a common base on which to manage the project.

(133) Software inputs to higher level management.

(133) Majority of problems seem to arise from lack of good historical data; speed of technology advance also complicates matters in this area.

(133) Require software management training for managers.

(133) Tight planning requirements.

(211) Software engineering projects can be planned using the same basic planning techniques as for any other project. This technique has worked for us on many programs.

(211) Management must understand the technical requirements of the job. All unknowns should not be buried in software.

(211) Detailed software development plan reviewed by senior ADP staff.

(211) Identify all necessary tasks and allow contingency.

(211) PERT charts or equivalent; formal status reporting.

(211) Teach project managers how to plan better.

(211) Detailed management plan.

(211) Matrix solution.

(233) Increase experience data base with time. The biggest uncertainty is when the requirements a firm enough to translate into a computer program. Firm requirements ususally come late in the program development, hence making software planning uncertain. Extensive early problems simulation is required to make software planning reasonably accurate.

(233) Treated software like hardware.

(233) Software engineering projects must follow proven engineering standards.

(331) Force a complete schedule and work load estimate bi-monthly by program module/component.

(333) Planning can be a waste of time if the inputs (specifications) are not complete. One must verify specifications first for completeness.

(411) Develop plans.

(444) The problem is to keep the development according to plan, assuming it was realistic to start with. The biggest problem is unfrozen requirements/change traffic.

<u>QUESTION 14.</u> Problem - There is an inability to accurately estimate delivery time of a computer program.

<u>SOLUTIONS</u>

(111) Identify a plan of action, insure schedules are understood, staff to meet the schedule commitments and insure a buffer time span is included to account for unpredictable problems. Include use of overtime and multiple shift operation to meet schedules.

(111) Leave no major problem, but be willing to accept crabs on minor, but sometimes very time consuming problems.

(122) Improved software development techniques.

(133) Provide enough time. Define.

(133) Measurements of past to relate to future.

(133) The trick consists in developing a good set of specifications before actual implementation begins. This avoids schedule slippage by changes not planned for.

(133) I doubt it can be solved. Realistic rules for estimating times/effort might help. Greater involvement by programmers/ analysts in setting schedule is needed. Wait until job is well understood before establishing delivery date.

(133) Do it right! Train managers! I established visibility techniques and daily meetings and planning papers.

(133) (did) Establish data base of projects histories. Prepare software cost estimation procedures. Insist on detailed milestone schedule.

(133) Staff with proven, experienced, software engineers; and strive to solidify requirements early. Use multi-skilled software designers in the area of input/output software design.

(133) Staff with experienced personnel, periodic management reviews, detailed scheduling with work performance measurement.

(133) More detailed schedule preparation.

(133) We have gone to milestones two weeks or less -- no more code/debug, etc. Modularity seems to be the key.

(133) Spend more time on estimates. Base estimates on past experience. Allow for unforeseen problems. Monitor progress very closely.

(211) Rely on competence of individuals.

(211) Assuming the requirements are complete, it is not difficult to estimate the design and coding efforts quite accurately, and even testing to within 15 percent.

(211) Improve the completeness of program requirements statements.

(211) (would) Bottom-up, etc.'s, filtered through updated experience factor; realistic funding and staffing expectations were evaluated monthly.

(211) Guess again when specifications have been written.

(211) Experience an important factor.

(211) Use experienced people to make best guess.

(211) Strict adherence to all intermediate delivery dates and correctness of intermediate items.

(211) Increased knowledge of problem complexity.

(211) In the field today an accurate estimate can be made if sufficient time is devoted to finding a comparable system. However, frequently a low estimate is used just to get the project started with hope that more funds will come later.

(214) Guess based on experience and drive to that date by manning adjustments, etc.

(233) Make specifications as complete as possible; minimize change.

(233) More conservative estimates must be made; more factors and contingencies examined.

(233) This problem was relieved by structured design and implementation of functions via the "build" philosophy of integrated and tested functions.

(233) Used staff with previous similar experience.

(233) Recognize that this is not an accurate portrayal of the situation. I think the real problem here is the inability to base line requirements -- principally because the long software lead time forces the customer to start the software specifications prematurely.

(244) Spent extra free time.

(433) Realistic independently verified sizing of the work effort.

(444) Assuming a realistic plan and schedule, the main problem is to control changes so as to be able to keep to the schedule and negotiate new dates with change.

(455) I feel that a thorough analysis of emperical data will result in curves useful in accurately predicting completion and delivery of software of any complexity.

QUESTION 15. Problem - The ability to plan for resources, particularly the number of programmers required is poor.

SOLUTIONS

(111) Project difficulty estimates must be improved.

(111) Inspect the task requirements very closely to insure complete understanding. Include sufficient personnel to support design, code, test, and documentation requirements. Allow for use of overtime or multiple shift operations in establishing number of programmers required.

(132) Have access to large data base use guidelines.

(133) Do it right! Train managers! I established visibility techniques and daily meetings and planning papers.

(133) This problem may be resolved by structured design, programming standards and judicious hiring of programmers to implement the system to be built in the selected language.

(133) Better communication between management and technical requirements to justify additional resources and funds.

(133) Properly size the program workload.

(211) Use of known productivity standards, i.e., lines of code per programmer.

(211) Bottom-up etc.'s, filtered through updated experience factors realistic funding and staffing expectations were evaluated monthly.

(211) Plan on a very detailed level with a clear understanding of working conditions. Many interior milestones are required to maintain a programming schedule.

(211) If requirements are complete then the level of effort is not difficult to estimate as a function of schedule and level of personnel expertise.

(211) In depth scheduling.

(211) Increase staffing as the need arose.

(211) Spent extra free time.

(211) Use of management personnel who have control over programming resources.

(211) Use management personnel who have strong influence over programming pool personnel.

(211) Use/maintain measurements of productivity vs size and complexity.

(211) Spend more time on estimates. Base estimates on past experience. Allow for unforeseen problems. Monitor progress very closely.

(211) Increased knowledge of problem complexity.

(211) Rely on experienced managers.

(214) Residue of the "software is mystery" syndrome. Knowledgeable technical management is the answer.

(231) In the field today, an accurate estimate can be made if sufficent time is devoted to funding a comparable system; however, frequently a low estimate is used just to get the project started with hope that more funds will come later.

(233) Heuristic rules for effort estimation would help. Provide for continuing review of resource requirements.

(233) This aspect improved with experience, but is subjective; depends upon the ability and motivation of available programmers.

(233) (did) Establish data base of project histories. Prepare software cost estimation procedures. Insist on detailed milestones schedule.

(233) Modularity approach has helped here -- it is easy to see manpower loading by modular assignment of personnel.

(233) Used staff with previous similar experience.

(311) Train and orient programmers to be flexible in assignment so they can be moved around more easily and effectively.

(311) Usually over estimate and under staff.

(444) Plan "smooth" manloading profiles, use previous similar projects as models, provide contingency plans.

(444) No problem.

(455) A thorough analysis of emperical data will result in curves useful in accurately predicting completion and delivery of software of any complexity.

(555) The real problem is making sure the right programmer is working on the right problem. This project has a tendency to keep a constant staff with a variety of talent so the original question does not apply -- none-the-less, matching talent to task is the real problem.

QUESTION 16. Problem - There is no real quality method of designing a project control plan that will enable project managers to control their project.

SOLUTIONS

(111) Follow specifications and continue reviews. Also stressing problem areas.

(111) Project manager's program plan must include project organization structures, reporting levels and milestones to be met. This should have concurrence of each functional group involved.

(111) (did) We have subjectively defined parameters that we wish to monitor and control, and have designed a scheme to do so.

(111) Reorganize project as soon as possible to give manager more control.

(111) Use automated project control systems (such as PCS-70 or MK-III) to control projects.

(111) New manager.

(111) Closely track manhours expended vs budgeted (CSCS/R, CPR) and compare with documented evidence of CPS design development progress.

(113) Assign more technically knowledgeable people to monitoring and enforcing the plan.

(133) Set up guidebook organizations.

(133) Project control may be accomplished by establishing an achieveable goal prior to designing the system. Milestones and schedules must be monitored continuously and managed.

(133) The milestone approach is OK with some additional identification of dependent tasks. "Pert" or similar monitoring tools are excellent but must be generated by technically knowledgeable people and thus cost a great deal.

(211) (would) Institute an earned value type scheme with meaningful work measurement parameters.

(211) Plan on a very detailed level with a clear understanding of working conditions. Many interior milestones are required to maintain a programming schedule.

(211) Perfection is not possible, but the critical pitfalls should be predicted.

(211) A plan must be flexible to change. It is only a plan. Control procedures are more important. The quality of the plan is usually the same quality as the manager who develops it.

(211) Develop plan that provides for continuing monitoring of tasks, status, schedule, etc.

(211) PERT charts or equivalent; formal status reporting.

(211) Educate management!

(211) Milestone schedules.

(211) Experience of the individual project manager appears to be major factor to success -- again, it's not to the "cook book" stage.

(233) Problem is really the putting of numbers on subjective measurements, (e.g., is to know your people).

(233) Do it right! Train managers! Established visibility techniques and daily meetings and planning papers.

(311) Develop a plan whereby project management does control project.

(333) Provide visibility.

(444) Frequent status reviews and customer involvement, approval of intermediate milestone completion, and affect change management are critical.

(444) I think there are quality methods of controlling projects -- PERT, WBS, walk-through's, MIS, . . .

(455) No problem, once the need and method of getting tangible results is understood.

QUESTION 17. Problem - There are no decision rules for the selection of management techniques for software engineering project management.

SOLUTIONS

(111) Cut and try helped. Put out management standards and school managers in this!

(211) Institute an earned value type scheme with meaningful work measurement parameters.

(211) Software project management, unlike many people's beliefs, is not considerably different than other engineering tasks, therefore similar rules may be applied.

(211) Management is not science. Many techniques are valid. One must select managers who have previously shown insight and talent on similar jobs.

(211) More research should be done in this area.

(211) (did) We have subjectively defined parameters that we wish to monitor and control, and have designed a scheme to do so.

(211) No rules are required, however, the proper management technique should be applied based on the particular project (i.e., matrix management where a project cuts across many disciplines, line management where a project is local to a line supervisor, etc.).

(225) Do not know an adequate estimating technique.

(233) Management techniques vary with the type of system to be built. The process should be as simple as possible. Proper planning between managers and technicians is identical to solving this type of problem.

(311) This is a question of philosophy. I do not believe that software engineering projects require applications of management principles which do not already exist.

(311) Management science guidebooks.

(311) Review alternatives and select techniques most applicable for project and project personnel.

(311) Must choose one and start with it.

(311) Develop base line of techniques in control plan.

(311) Tried new method to see if it would work (it didn't!).

(311) Consider previous similar efforts and work to solve specific problems experienced and duplicate successes.

(333) Learn what is available and make rational decisions on a case by case basis.

(333) Treat each case individually.

(344) There is nothing unique about software that requires special techniques.

(411) For flight software the techniques are by largely unique to the project and the customer requirements. Decision rules should be flexible to be responsive to this. Good management judgement will suffice for rigid decision rules.

(444) Management techniques are the option of the software development company organization and the customer. There should be frequent continuous realistic sharing of experiences to identify effective techniques.

(444) I think there are quality methods of controlling projects-- PERT, WBS, walk-through's, MIS, . . .

(455) There appears to be a multitude of management techniques. It's a matter of selecting those which work within your particular environment.

(455) CPLSS computer program development plan.

(455) You can't have rules for everything.

(455) Experience.

QUESTION 18. Problem - Techniques for the selection of project managers are poor, generally resulting in poorly managed projects.

SOLUTIONS

(111) (would) Select managers who have previously been highly successful in identical projects.

(111) Hire good ones, train/experiment with new ones.

(111) I don't think this is a problem here. If it were I would seriously consider changing employment.

(111) Insure potential project managers have the necessary background (technical and management) and have strong personalities and can deal with both management and technical personnel. Also insure they have good communications skills and can deal with customers effectively.

(111) Qualifications must be experienced based.

(133) Must examine past projects to see if any trends exist. Look for some correlations between qualifications and successes.

(133) Select project managers based on experience in related areas to insure that the risk is reduced to the lowest possible level.

(211) Selection must be based on ability in terms of managerial and technical talent, rather than availability of an individual.

(211) Better matching of project specifications to project manager background.

(211) Management is not a science. Many techniques are valid. One must select managers who have previously shown insight and talent on similar jobs.

(211) (would) Define the attributes of a good manager and match those attributes to the capabilities of potential managers. Develop an apprenticeship program for project managers.

(211) Very hard to solve.

(211) Extensive interviewing, test his technical ability.

(211) Emphasize track record.

(211) The biggest problem I've witnessed here is pushing technical people into management slots they don't want and/or not prepared for.

(211) Try out potential project managers on small projects first.

(222) Very hard to solve. Project managers with applicable experience should be assigned.

(233) Give senior analysts responsibility and accountability whenever possible, involve programmer/analysts in planning to train them as future project managers.

(241) Disagree. Project manager should be selected from group with proven record of success.

(311) Based upon previous performance.

(311) I'd say we're not doing too bad in this area.

(311) By bringing them through the ranks as they qualify.

(411) Technique other than management judgement based upon demonstrated skill and satisfactory performance.

(411) Develop managers over a period of time.

(444) Assure a pool of trained, experienced project managers exist; conduct courses on useful management techniques, make sure reviewing authority is a good manager.

(455) My experience is that manager selection is fairly good.

(511) Disagree with problem statement, but can be overcome by flagging the problem areas immediately.

(555) Should be knowledgeable in the discipline. Same criteria as selecting any other project manager.

QUESTION 19. Problem - There is no me:ns of measuring with any degree of accuracy the quality of code produced by a programmer.

SOLUTIONS

(111) Monitor the performance of the man's functional development and classify him on the degree of success of the function.

(133) Use modern programming techniques. Much emphasis on testing.

(133) Define problem. Set up measurements.

(211) Establish and enforce standards together with reviews and walk-throughs.

(211) Set up review process in which other programmers disk check code.

(211) Initiate quality control measures to check all code prior to release to the customer. The QA function should be independent from the project management and have the authority to veto the release of all code that does not meet required standards.

(211) Keep records of quality (e.g., errors discovered later, subjective evaluations, etc.).

(211) Management must know their programmers abilities.

(221) Get good programmers.

(222) More effort should be spent in code optimizatiou.

(222) Hire good ones; train/experiment with new ones.

(222) Code authorization should be one of the tasks.

(222) (would) Code must at least meet requirements and follow standards (structuring, naming, etc.). Continue to investigate current studies of design-versus-"quality" parameters (e.g., error rate versus cyclamatic complexity) and adopt those which are valid and fit within program constraints.

(222) Require testing milestones.

(222) (did) Separate test team. Extensive test.

(233) Establish and enforce coding standards, meaningful.

(233) Some standards could be established; e.g., ratio debugging to coding time, percent of original statement replaced, average statement complexity. Provide for code review by experienced programmers who know efficient, maintainable code.

(233) (would) Metrics of "good code" must be defined.

(233) Measurement is not as important as visibility into the code and a comprehensive test cases(s). Any measurement would be subjective since human beings are involved rather than machines.

(233) Measurement of programmers work can be measured by monitoring problem reports, computer usage and programmer documentation.

(233) Efficiency may be measured by insuring adherence to programming standards, early unit test procedures and system performance monitors.

(233) Peer group review.

(311) More frequent review of programmer work.

(311) Code has to be examined by head programmer or project managers, and some judgement made.

(322) If you mean "in advance" I agree. This is an important problem. If you mean during testing and early operation, I think we can tell good code and good programs from bad.

(322) What is accuracy? Code walk-through's?

(333) Not a critical problem; i.e., the code does the job. Prefer to be "more people oriented" and not put people into a 2,000 or 10,000 "line of good code per year" box.

(333) Sequential software testing does measure code quality, software techniques of programming standards, walk-throughs, automated checkouts, etc., all help.

(333) There are methods, but they cost money because they require a certain duplication of effort.

(455) IBM seems to have made great strides in this area. Problem I see is that not too many organizations can afford the overhead required to do tracking at this level.

(455) Given appropriate measures of "goodness", goodness can be measured. Is all this effort worth the trouble and cost, or would the time and money be better spent on definition of standards which produce known results?

(555) Hire high quality coders if that was my primary concern, I'd do this as soon as someone defined "quality of code."

QUESTION 20. Problem - There is a poor accountability structure in most development projects, leaving some question as to who is responsible for various project functions.

SOLUTIONS

(111) Establish procedures and establish good work breakdown structure.

(111) Responsibility should be clearly defined.

(111) Streamline accountability structure.

(111) Couple responsibility with accountability; make sure tasks are understood by all parties.

(111) Establish clear-cut lines of responsibility for each project function and organize the project around the functional requirements. Insure all project members know who is second in command for each major function under the project.

(111) Strict monitoring of functions and personnel is essential.

(133) The system should be designed in such a fashion that functions may be allocated to programmers for implementation, integration and testing.

(211) Update responsibility, documentaion, and organization charts.

(211) Clearly defined work packages.

(211) (did) Enforcement of the programming team concept.

(211) Document task assignments.

(211) Invariably the informal organization is predominant. Responsibility is important only when a job is not getting done. Responsibility should be given to a person who is most capable of performing.

(211) (did) Project organization with written tasking and procedure direction.

(211) Completely documented and reviewed task descriptions with agreed upon schedules and outputs.

(211) When projects are specified, each task should be outlined and lines of responsibility identified.

(211) Clearly define and then respect the accountability structure.

(211) This problem is eliminated by the development of good program plans at each level, including specific personnel assignments and responsibilities.

(211) Define responsibilities in a better way.

(211) Programming by contract -- i.e., the team members would sign contracts that specify their goals and responsibilities.

(311) Use standard WBS structure.

(311) Assume the responsibility if no one else does.

(311) Clearer definition of responsibilities.

(311) Task descriptions, organization charts.

(333) This is a planning problem and can be minimized if the planning group consists of technically capable people, who are given sufficient time to identify all functions in the initial plan.

(411) Matrix organization.

(444) Clear organization structure, good work authorizations usually keep this under control.

(455) Surface it in the WBS.

(555) I don't truly agree with this statement. Responsibility is generally well identified, although not always in the proper location.

QUESTION 21. - Problem - There is much consternation in industry concerning how best to organize for the accomplishment of a project. (e.g., Should the project be organized around the function, the project, or under a new matrix system?)

SOLUTIONS

(111) Select the system that best suits the particular project. No one system will respond to the needs of all types of projects.

(111) Work independent as though it was a line organization for the one project.

(111) There is no single organization good for all projects. At high levels, some form of matrix to get the proper people is necessary. However, a second level of organization within the project is important and this level must be geared to the specific project guide.

(111) Change companies. I give up here.

(133) This problem may be relieved by organizing the project to meet the needs of the system by coordinating management and technical planning effort.

(211) Better evaluate the specific project objective and then apply the organization that seems to better fit.

(211) Matrix system dictated by division management.

(211) New matrix system.

(211) Organize by function.

(211) Pick best way in keeping with company dynamics.

(211) New matrix system.

(211) (did) Project organization resulted from a year of trial and error.

(211) I don't know of anyone that seriously believes he can solve this problem.

(211) Project or matrix organization.

(211) Matrix organization with emphasis on project management.

(211) Selectively.

(233) For projects which fluctutate in manpower requirements, matrix is clearly the best approach. From a technology viewpoint the individuals need to see a clear, career path where specialization is allowed. Long term projects, where manpower requirements exist full time, the projectized organization is more appropriate.

(233) Depends on your particular organizational/customer mix as to which works best.

(233) Matrix system.

(311) Have clear set of criteria in the beginning as to what are important values assigned to development organization, and pick the approach to best satisfy project values within company environment.

(311) This depends on overall company structure policy. Each type of organization has its pro's and con's. Each will work if there is a dedication by management to enforce standards, planning, reviews, etc.

(311) Select one type and adapt it to particular needs, use it enough to make it the normal method of development.

(333) All methods work; the best way is to determine case-by-case.

(411) Organize to suit the project/problem to be solved. No one way will solve all situations.

(444) Project organization.

(455) There are many ways to skin a cat. The program in which the boss is most interested will always get the best resources.

(455) The scope of the job dictates the answer.

QUESTION 22. Problem - It is difficult to impossible for a project manager to have the requisite visibility to be able to determine whether the project is on schedule and within cost.

SOLUTIONS

(111) Establish proper visibility tools, seeing papers at conferences.

(111) Better reporting techniques need to be developed.

(111) (did) Developed and automated reporting procedures that reports status of all software modules. Developed cost/performance chart for earned value reporting.

(111) Insure that only responsible persons are generating status reports, problem reports, budget reports, etc. Utilize automated accounting systems and project control systems whenever possible to insure timeliness in reporting schemes.

(111) The manager must go after information, "Look under the rug," and provide help where needed. Further, he must keep the customer informed before dates are missed.

(114) 1. Good spec's. 2. Tangible milestones. 3. Reviews, releases, audits. People hate documentation (unless they either don't have the opportunity to learn to hate it, or they see what a beautiful thing it can be). But getting the documentation <u>integrated</u> into the design process essentially clears half of the "problems" you cite.

(133) Need cost/accomplishment matrix.

(133) 1. Good task leaders. 2. Formal reporting structures. 3. Review boards. 4. Formal sufficiency and acceptance criteria.

(133) This problem may be solved by enforcing scheduled deliveries and comprehensive testing procedures.

(211) Define program functions at a detailed level and use that breakdown as a check-off list as tests are completed.

(211) By establishing target dates and budgets, a project can be controlled and evaluated at any point.

(211) Such visibility is a function of control structure established, reporting tools used, clear-cut level of responsibility, and continued personnel involvement.

(211) If he doesn't the project manger should be replaced. His job is to maintain just such visibility.

(211) Define multiple intermediate milestones with observable, measureable outputs, e.g., specifications with walk-throughs.

(211) Use rate charting for visibility of module development.

(211) Weekly progress meetings.

(233) Have a separate cost control function.

(233) Better management information systems, built around automated data acquisition and utilizing automated work breakdown structures.

(411) (did) Thorough planning, project organization, reporting by exception, tight change control, frequent reviews.

(411) Used matrix system.

(444) Milestones.

(444) Good budgets, frequent control is essential. The recent C-specifications and CSSR techniques applied to software development, detract, rather than help.

(533) Plan, control, and measure against the plan. You can't answer these kind of pseudo motherhood questions.

(555) I don't agree here. Modularity has given us the visibility. Improved communications and fixing of responsibility.

(555) Disagree -- use many milestones or kilometer stones.

QUESTION 23. Problem - There is a general lack of traceability from the requirements specification to the final code.

SOLUTIONS

(111) More frequent reviews are needed.

(111) (did) Traceability matrix from performance specifications to design specifications.

(111) Current contract requirements remove this as a problem since part of preliminary and critical design reviews relate the specification requirements to the actual code that implemented the requirement.

(111) Requirements should lead to the development of performance specifications which leads to design specifiations. These documents must iclude detailed test plans for each functional area. So that the coder can code to the design specifications, mindful of test cirteria.

(122) Formal requirements methodology; automated documentation.

(122) Correct through documentation.

(123) Add integrated approach to software development and testing is required.

(133) Threads, SUD's and similar decomposition techniques, where the decomposed elements could be weighted, tracked through implementation and integration, back to complete functions.

(133) This problem may be relieved by enforcing structured design, programming standards, test procedures and documentation standards.

(133) It is only through traceability that the accuracy of the final product can be determined relative to performance requirements. Traceability absolutely necessary.

(133) The requirements specifications should be the basis for final acceptance test procedures, which are best written by someone not involved with the original specifications.

(211) Traceability matrixes from document to document including code.

(211) Insist on traceability.

(211) Require high traceability.

(222) Better analysis of requirements.

(222) Provide good documentation and tacking mechanisms to each lower level of design detail.

(223) Accurate specifications are required with any changes.

(223) Good old V&V rigorously applied and independently performed. Double the traceability for double the development cost.

(233) Do this by edict and by methodology.

(233) Enforce adherence to both project specification and design standards.

(233) Rigorous test procedure development.

(233) Again, I feel this is a matter of overhead cost. A number of these items are, in fact, niceties rather than hard requirements.

(233) Traceability matrices at each level.

(233) Project review detailed design with specific intent to trace requirements through design.

(233) (did) Extensive test by a separate group.

(233) Set up guidelines.

(311) Clear assignment of responsibilities and documentation of technical decisions.

(311) Improved documentation of various phases.

(333) Some new automated tools which generate and evaluate the matrix helps. Writing separate testable requirement helps, intentionally organized design also helps.

(333) Tight control.

(455) A technical control board at the project level.

QUESTION 24. Problem - There is, in general, an inability to measure the quality of the program.

SOLUTIONS

(111) Better testing.

(111) Organized around a systemized project.

(122) Different programs can be tested through different means to measure its quality. For instance, by using certain input (with results being known in advance) to test the program and compare its outputs with the result to determine the program accuracy. The compute time is another way to determine the factor of quality.

(122) Develop software metrics.

(133) This is a true statement -- unfortunately, programming is like a good landing -- anyone you can walk away from.

(133) Define criteria and weapons software.

(211) Assure that the program has followed all design specifications and implementation standards.

(211) Control/quality meetings.

(222) (would) Establish a realistic MTBF criteria; measure against requirements and standards; impose standards to support LSS (structuring, naming, etc.).

(222) Do this by edict and by methodology.

(222) Still have not established a good measure of quality.

(222) (did) Extensive testing by separate group.

(233) Did not consider measuring the quality.

(233) Still working the problem.

(233) Some sort of programming standards could help. Review of programming (for efficiency, readability, modularity, etc.) while there is still time to change.

(233) The quantity of a program may be measured by way of test results, adherence to programming standards, structured design nd accuracy of interfaces.

(233) Improved testing.

(311) Due to high cost of superior quality software, accept software that meets user requirements, don't optimize, etc., or cost is too high.

(311) Hire high quality programmers.

(322) If you mean "in advance" I agree. This is an important problem. If you mean during testing and early operations, I think we can tell good code and good programs from bad.

(333) "Quality" is a subjective term. If the standards applied result in a program that meets the requirements, "works," and is maintainable, then its "quality" is satisfactory.

(333) Code has to be examined by head programmer or project manager and some judgement made.

(333) This is the function of the QA organization. Any mutually agreed to method which you can afford will work. Don't expect absolute definitions.

(355) Again, if it performs the requisite function, it may be used. Recognize the variants in programmer ability.

(355) Overcome by experience and by use of the program.

(422) Measurement qualification will always be subjective. A 90 percent figure is of no value, if the remaining 10 percent code of the program results in a systems failure.

(455) The measure of the quality of a program is best accomplished by demonstrating all requirements to user and measuring his response.

(455) "Quality" unless made more specific means little. "Quality" in what way: cheap, timely, fast . . .?

(555) Success criteria cannot be anything but meeting requirements. If not a requirement, then cannot be judged. Put it in the specifications, and test for it. Here is the crux -- if you can decide on a meaningful test, then you will get a meaningful design.

(555) I don't think the question is correct. I think it is possible.

ED

8